Universität Zürich[UZH]

Geographisches Institut

You are what you eat? - A comprehensive analysis of saprotrophic fungi and their substrates in the Laegern forest

GEO 511 Master's Thesis

Author: Ladina Reich, 18-734-400

Supervised by: PD Dr. Guido Lars Bruno Wiesenberg

Faculty representative: PD Dr. Guido Lars Bruno Wiesenberg

22.04.2026

Abstract

In forest ecosystems, saprotrophic fungi that feed on wood and bark play an important role in decomposing dead organic matter. Thereby, they make organic compounds available to other organisms and enhance C and nutrient cycling. Nevertheless, most studies have analysed either fungal tissues or substrates separately. As a result, the biochemical link between decomposing plant material and fungal biomass remains poorly understood. This thesis investigated decomposition as an integrated system by analysing three saprotrophic fungal species and their wood and bark substrates from the Laegern forest. Biological compartments (fungal tissues, wood and bark), rot types (white rot vs brown rot) and substrate types (hardwood vs softwood) were compared using elemental and organic chemical analyses, including fatty acid and sterol profiles. Fungal basidiocarps differed strongly from their substrates, showing higher levels of N, ergosterol, and fatty acids. Chemical differences were even found within the basidiocarp tissues, with higher N values in the hymenium and higher C values in the pileipellis. Because ergosterol was found in fungal and plant compartments, fungal presence was observed in wood and bark substrates. In contrast, plant sterols were only detected in the substrates. Wood was characterised by lignin-rich compounds and low N concentrations, whereas bark contained higher N, fatty acid, and sterol contents. These findings suggest that fungi are not just a reflection of their substrates but rather transform plant material into fungal biomass. Rot decay differences were reflected in the substrate, with higher lignin-like residues in softwoods and higher cellulose concentrations in hardwoods. In contrast, differences between basidiocarps of *F.pinicola* growing on hardwood and softwood were comparatively small.

Zusammenfassung

In Waldökosystemen spielen saprophytische Pilze, die sich von Holz und Borke ernähren, eine wichtige Rolle beim Abbau toter organischer Substanz. Dabei setzen sie organische Verbindungen frei, die anderen Organismen zur Verfügung stehen, und fördern damit den Kohlenstoff- und Nährstoffkreislauf. Die meisten Studien untersuchten jedoch entweder Pilzgewebe oder Substrate separat. Dadurch bleibt die biochemische Verbindung zwischen zersetzendem pflanzlichem Material und pilzlicher Biomasse bislang nur unzureichend verstanden. In dieser Arbeit wurde die Zersetzung als integriertes System untersucht, indem drei saprotrophe Pilzarten und ihre zugehörigen Holz- und Rindensubstrate aus dem Lägern Wald analysiert wurden. Biologische Kompartimente (Fruchtkörper, Holz und Rinde), Rottypen (Weissfäule vs. Braunfäule) sowie Substrattypen (Hartholz vs. Weichholz) wurden mittels elementarer und organisch-chemischer Analysen, einschliesslich der Fettsäure- und Sterolprofile, verglichen. Die Fruchtkörper unterschieden sich deutlich von ihren Substraten und wiesen höhere Stickstoff-, Ergosterol- und Fettsäuregehalte auf. Auch innerhalb der Fruchtkörper wurden chemische Unterschiede festgestellt, mit höheren Stickstoffgehalten im Hymenium und höheren Kohlenstoffgehalten im Pileipellis. Da Ergosterol sowohl in Pilz- als auch in Pflanzenkompartimenten nachgewiesen wurde, konnte Pilzbefall in Holz- und Rindensubstraten beobachtet werden. Im Gegensatz dazu waren pflanzliche Sterole auf die Substrate beschränkt. Holz zeichnete sich durch ligninreiche Verbindungen und niedrige Stickstoffkonzentrationen aus, während die Rinde höhere Stickstoff-, Fettsäure- und Sterolgehalte aufwies. Diese Ergebnisse legen nahe, dass Pilze nicht nur ein Spiegelbild ihrer Substrate sind, sondern vielmehr pflanzliches Material in pilzliche Biomasse umwandeln. Unterschiede beim Fäulnisabbau spiegelten sich in den Substraten wider, wobei Weichholz höhere Anteile an ligninähnlichen Rückständen und Laubholz höhere Zellulosekonzentrationen aufwies. Im Gegensatz dazu waren die Unterschiede zwischen Fruchtkörpern von *F.pinicola*, die auf Hart- bzw. Weichholz wuchs, vergleichsweise gering.

Table of Contents

I. List of Figures

II. List of Tables

III. List of Abbreviations

Aromatic C=C	Aromatic carbon-carbon bonds
Aromatic C-H	Aromatic carbon-hydrogen bonds
Aliphatic C-H	Aliphatic carbon-hydrogen bonds
a.s.l	above sea level
AUC	Area under the curve
BCFA	Branched-chain fatty acids
BR	Brown rot
C	Carbon
Carboxylic C=O	Carboxylic carbon-oxygen bonds
DCA	Dicarboxylic acids
DRIFT	Diffuse Reflectance Infrared Fourier Transform
EA-IRMS	Elemental Analysis – Isotope Ratio Mass Spectrometer
FA	Fatty acids
FID	Flame ionisation detector
F.pinicola	*Fomitopsis pinicola*
HW	Hardwood
IS	Internal standard
LCFA	Long-chain fatty acids
LMEM	Linear Mixed Effects Model
MS	Mass spectrometry
N	Nitrogen
PCA	Principal component analysis
SCFA	Short-chain fatty acids
SSCFA	Saturated short-chain fatty acid
SLCFA	Saturated long-chain fatty acid
SW	Softwood
TC	Total Carbon
TLE	Total lipid extract
TN	Total Nitrogen
USCFA	Unsaturated short-chain fatty acid
ULCFA	Unsaturated long-chain fatty acid
WR	White rot

1 Introduction

1.1 Fungi

Forests are complex ecosystems that support a high diversity of organisms such as plants, animals and microorganisms (Sudharsan et al., 2023). Within these systems, fungi play an important role in C and nutrient cycling by both promoting the formation of woody biomass and driving its breakdown through biological decay (Barron, 2003). This means that fungi do not grow in isolation; instead, they form complex interactions with their surroundings that play a crucial role in terrestrial biogeochemistry. To better understand their ecological importance in forest ecosystems, it is essential to gain a deeper understanding of their chemical composition, morphology, and nutrition.

Fungi are a diverse group of eukaryotic, multinucleated organisms that form the kingdom *Fungi*. There exist in total four phylogenetic groups of fungi: *Phycomycota* (lower fungi), *Ascomycota* (sac fungi), *Basidiomycota* (club fungi) and *Deuteromycota* (fungi imperfecti). The two major phylogenetic groups of fungi are *Ascomycota* and *Basidiomycota*, which are distinguished by the way they reproduce (Singara Charya, 2015). Fungi active in the wood decomposition process include mainly basidiomycetes (Grinhut et al., 2007). Despite interspecific variation in elemental composition, fungi generally contain around 50% of carbon, 35% oxygen, 8% nitrogen and 6% hydrogen by mass. Their cell walls are primarily made up of carbohydrates (polysaccharides such as glucans and chitin), proteins, moisture, ashes and lipids (Dore et al., 2014).

Fungal spores can be blown in the wind or carried (e.g. by insects) to the wood and enter it through wounds on the tree trunk, tree stem or dead roots (Klug & Lewald-Brudi, 2023). As hyphae come into contact, they start to fuse (hyphael fusion) and form a mycelium, an interconnected network of filaments. The hyphae secrete enzymes into the substrate, which breaks down biological polymers into monomers. These are then absorbed into the mycelium by facilitated diffusion and active transport. The mycelium serves as a reservoir and distribution system for nutrients. Mycelial growth releases CO_2 back into the atmosphere (Watkinson et al., 2006). Fungal mycelia can become visible, for example, on various substrates in the forest. Once the mycelium reaches sufficient size, the fungus produces a fruiting body, also called a sporocarp, for spore dissemination, typically composed of a stipe, cap, and gills (Figure 1). The sporocarp of a basidiomycete is called a basidiocarp. Basidiomycetous fungi often produce large basidiocarps in the later stages of wood decay processes (Goodell et al., 2008). The basidiocarp can further be separated into pileipellis (cap surface), trama (structural flesh), and hymenium (fertile layer) (Schultz et al., 2014) (Figure 1). The pileipellis is the uppermost layer in the pileus (cap) of a fruiting body. It serves as a protective layer covering the fleshy tissue (trama) (Synytsya et al., 2023).

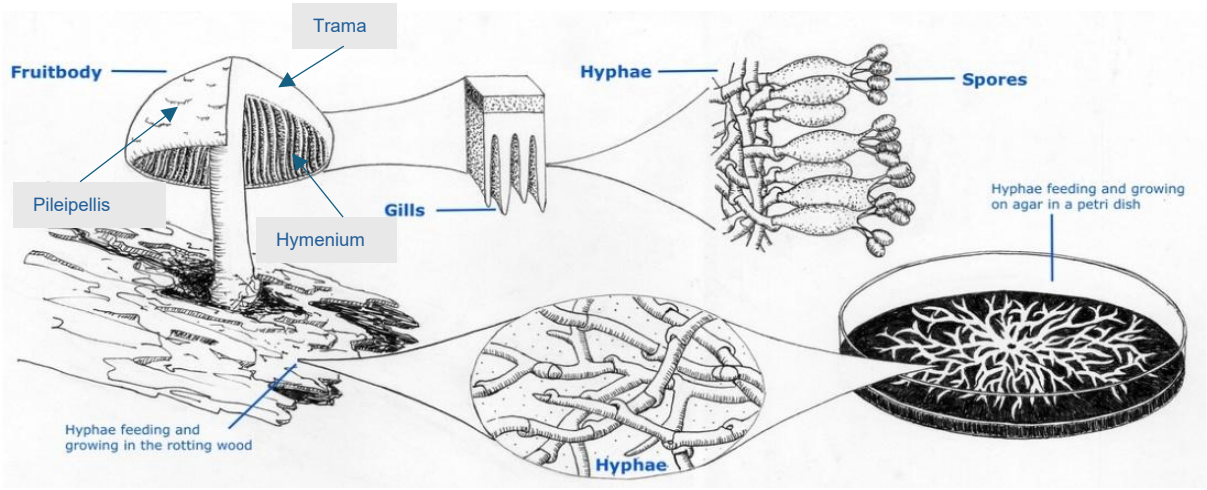

Figure 1: Hyphae feed, grow and branch to form a colony and a fruitbody. Adapted from Science Learning Hub – Pokapū Akoranga Pūtaiao, by Manaaki Whenua – Landcare Research, 2016.

As heterotrophic organisms, fungi obtain nutrients by secreting digestive enzymes into their environment and absorbing these organic molecules (Gadd, 2013). Relying on external organic matter for nutrients, fungi have evolved diverse strategies for resource acquisition. In total, there are four different types of nutrition: parasitic, predatory, mutualistic, and saprobic (Singara Charya, 2015). Parasitic fungi absorb nutrients from the living cells of the host organism and grow inside the host organism. Saprotrophic fungi are the largest group and are called decomposers because they feed on dead organisms and organic waste. In forest ecosystems, saprotrophic fungi are considered the most efficient decomposers. They act as primary, secondary and tertiary decomposers, which recycle large amounts of C as well as other nutrients (Grinhut et al., 2007). Based on the substrates saprotrophic fungi decompose, they can be further categorised into: Wood-decomposing fungi, Litter-decomposing fungi, Soil-decomposing fungi and fungi decomposing animal faeces (Eichlerová et al., 2015). Some fungal species start growing as parasites on living trees and change their nutrition to saprotrophic as soon as the host substrate dies (Li et al., 2022).

1.2 Wood and bark as fungal substrates

Wood formation supports the structural development of forest trees (Moron et al., 2025). The tree trunk features three main sections: the heartwood, which is physiologically inactive, the sapwood, which actively transports water and nutrients, and the bark, which protects the interior of the tree trunk against microbial degradation (Figure 2) (Moron et al., 2025; Ristinmaa et al., 2024; University of cambridge, 2004).

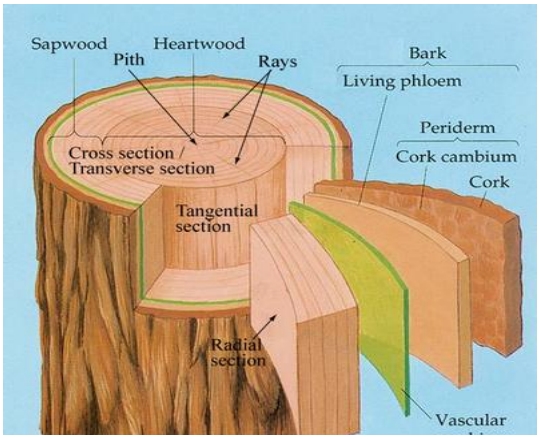

Figure 2: Schematic transversal cut of a tree trunk. Reprinted from The structure of wood (II), by DOITPoMS, University of Cambridge, 2004.

Forest ecosystems store a major portion of organic C as wood. Deadwood, consisting of standing dead trees, logs, snags and stumps plays an important role in nutrient availability, biogeochemical dynamics of key elements and soil development in forest ecosystems (Bani et al., 2018; Micó et al., 2024). Wood and bark comprise approximately 50% C, 44% oxygen and 6% hydrogen, with small amounts of nitrogen and inorganics (Moron et al., 2025). The major chemical component of a living tree is water, but on a dry-weight basis, all wood and bark cell walls consist mainly of carbohydrates (65-75%) that are combined with lignin (18-35%) (Stamm 1964 in Rowell, 2012) (Figure 3). Lignin is the second most abundant biopolymer in nature and is built mostly from aromatic compounds (Lubbers, 2025; Waggoner et al., 2015). The major carbohydrate portion of wood is composed of the polysaccharides cellulose (44-55%) and hemicellulose (20-30%) (Stamm 1964 in Rowell, 2012). Hemicellulose and lignin together create a protective barrier around cellulose. The structure of the so-called lignocellulose (lignin, cellulose and hemicellulose) is compact, with different bonding among cellulose, hemicellulose and lignin that makes lignocellulose a very complex substrate for fungal enzymes (Andlar et al., 2018). Wood further consists of extractives (4-13%) and ash (less than 1%) (Figure 3). Wood extractives are lipophilic substances including triglycerides, fatty acids, waxes and sterols. Although extractives are a minor component of wood, they play an important role in wood protection. Ash contains minerals such as calcium and potassium (Moron et al., 2025). Bark extractives such as fatty acids, alcohols, resins, pigments and tannins make up about 20%-40% of bark dry mass, while the suberin, lignin and phenolic acids form the remainder (Dossa et al., 2018).

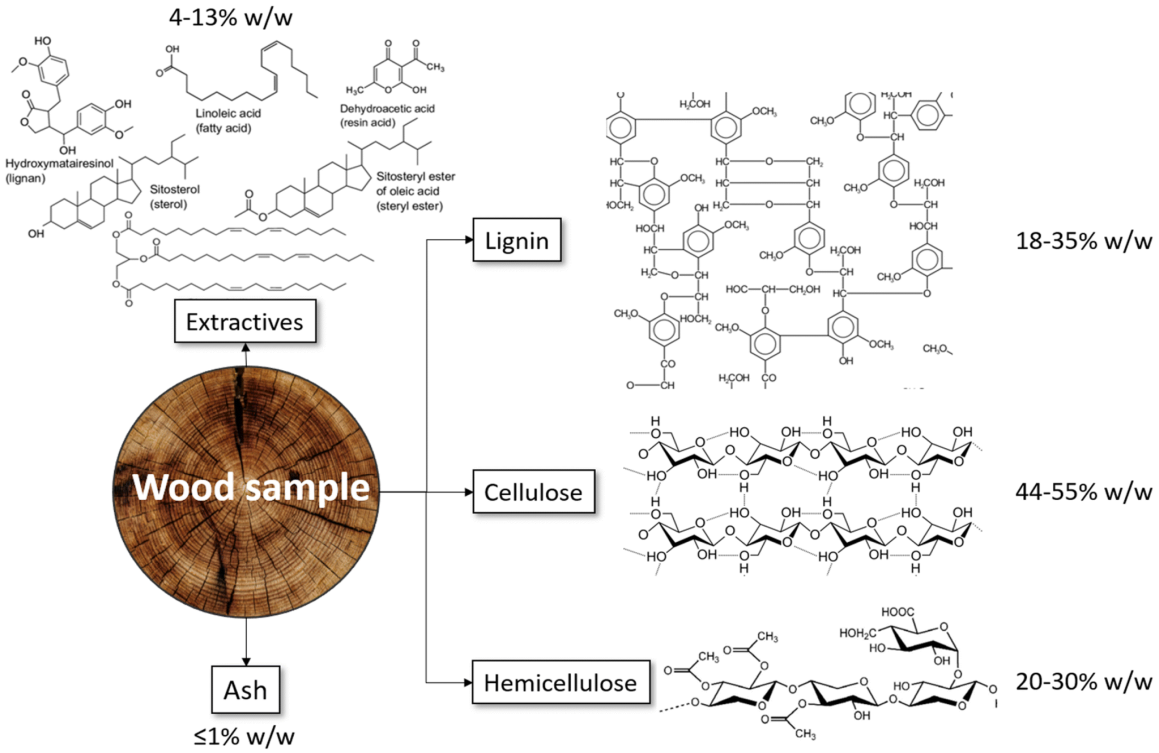

Figure 3: Chemical composition of wood (%w/w, equivalent to wt%). Reprinted from RSC Sustainability, by Moron et al., 2025.

Wood can be classified into softwoods and hardwoods, distinguished by their cellular structure (Moron et al., 2025). Coniferous trees (e.g. pine, spruce) are softwoods and broad-leaved trees (e.g. beech, oak, birch) are hardwoods. Hardwoods and softwoods show different compositions and structures of cellulose, hemicellulose and lignin (Rese et al., 2025). The higher lignin content in softwoods contributes to their strength and durability. Therefore, its density decreases more slowly than that of hardwood trees. Hardwood trees fully decompose in a rather short time period (around 25-40 years) compared to softwoods (Bani et al., 2018). In general, softwoods have slightly higher extractive amounts but lower ash content than hardwoods (Rowell, 2012).

1.3 Fungal mechanisms of wood and bark decay

In natural forest ecosystems, a dynamic equilibrium exists between the accumulation and breakdown of woody biomass (Schwarze, 2007). The breakdown of wood and bark compounds occurs by decay, rot and decomposition (Srivastava et al., 2013). Decomposition refers to the process by which dead organisms tissues break down into simpler forms of matter. This breakdown is very important as it is the basis for recycling limited chemical compounds, as well as freeing up limited physical space in the forest environment (Swift, 1977). Wood decay, for example, refers to the gradual deterioration or decomposition of wood over time, while rot specifically refers to the decomposition of organic matter caused by bacteria or fungi (Srivastava et al., 2013). The group of wood-decaying fungi is one of the few organisms capable of chemically breaking down wood (especially its component lignin) and thereby obtaining food and energy from it (Srivastava et al., 2013). As primary decomposers in forest ecosystems, they play a key role in returning C to the soil or atmosphere and releasing nutrients to the environment for reuse by other organisms (Jones et al., 2020; Li et al., 2022). The term “wood decay fungi” does not refer to a specific species or taxonomic classification but instead serves as a broad term encompassing a collection of fungi.

The ability of saprotrophic fungi to modify and degrade lignocellulose (lignin and cellulose) in wood and bark relies on both enzymatic and non-enzymatic systems (Andlar et al., 2018). Although most wood decaying fungi produce polysaccharide-degrading enzymes, including cellulase, xylanases and mannanases, these enzymes can not effectively penetrate or degrade woody substrates while lignin remains intact. In plant cell walls, lignin forms a protective matrix surrounding cellulose, which must first be modified or partially removed to allow access for cellulose hydrolysis (Blanchette et al., 1985).

On wood, decomposition is recognisable by visual detection of mycelium or by a typical decomposition pattern. Based on this pattern, fungi can be categorised into white-rot fungi, brown-rot fungi and soft-rot fungi (Li et al., 2022). Since this thesis focuses on brown-rot and white-rot fungi, soft-rot fungi will not be discussed in more detail. White-rot and brown-rot fungi are predominantly members of the Basidiomycota (Goodell et al., 2008).
Although lignin is an enormous natural C reservoir, only a small group of basidiomycete fungi, namely white-rot fungi, have evolved the ability to depolymerise lignin efficiently (Del Cerro et al., 2021). White rotted wood is pale in colour, soft and fibrous due to the lack of brownish

lignin (Mali, 2021) (Figure 4). White-rot fungi decompose and modify all wood components via enzymatic and non-enzymatic systems. For example, lignin peroxidase and manganese peroxidase, enzymes produced by white-rot fungi, are involved in lignin degradation. The most efficient lignin degraders are white-rot fungi that occur in dead hardwoods, although they can also infect softwoods (Baldrian, 2017). Up to date, it is debatable whether white-rot fungal species can utilise lignin as a C source for themselves or if they degrade it to access cellulose and hemicellulose (Camilleri et al., 2024; Del Cerro et al., 2021).
The wood decayed by brown-rot fungi is generally brown and crumbly, and it cracks in a cube-like manner (Figure 4). The brown colouration is mainly a result of modified lignin and the accumulation of iron in the decomposed wood (Mali, 2021). Brown-rot fungi can modify lignin but cannot decompose it. Because brown-rot fungi have genetically lost a few enzymes, they evolved non-enzymatic approaches to decompose wood components (Goodell et al., 2008). They primarily decompose cellulose and hemicellulose non-enzymatically via an oxidative process called Fenton chemistry, which produces hydrogen peroxide during hemicellulose breakdown (Srivastava et al., 2013). While white-rot fungi are more commonly found on hardwoods, brown-rot fungi occupy a different ecological niche and are mainly associated with softwoods (Xue et al., 2025).

Saprotrophic wood-decay fungi can also be classified by the range of substrates they colonise and decompose. Species that are restricted to a single type of wood are referred to as specialists, whereas fungi capable of decomposing multiple wood types are called generalists. Both white-rot and brown-rot species include specialist and generalist species capable of colonising hardwood and softwood species (Xue et al., 2025).

Figure 4: White rot decay of hardwood and brown rot decay of softwood (from left to right). Note. Created by the author.

1.4 Research gap

Determining fungal biomass in wood has long been challenging because fungal hyphae grow and spread within the substrate. As a result, fungal mycelia are not easily isolated from the wood and bark tissues by either optical or mechanical methods. One way to circumvent this problem is to quantify a cell constituent that occurs in relatively constant amounts in fungal tissues (Gessner, 2020). Ergosterol, as a major membrane component, is the main constituent that determines metabolically active fungal biomass, as it is found almost exclusively in fungi (Gessner, 2020; Mille-Lindblom et al., 2004). In this thesis, ergosterol concentrations will be measured to assess the contribution of fungal activity in deadwood.

For many years, the natural abundance of ^{15}N and ^{13}C has been investigated as a potential indirect method for assessing the functional roles of fungi in forest ecosystems (Gadd et al., 2007). Stable isotopes of C and N can reveal whether fungal tissues incorporate substrate-derived elements or whether fractionation occurs during fungal metabolism. If saprotrophic fungi preferentially utilise specific substrates, isotopic signatures of wood and bark should be reflected in fungal tissues. For example, when saprotrophic fungi are specialised or show a preference for particular substrates that differ in isotope ratios, then this could be reflected in the values in the mycelium or the basidiocarp (Gadd et al., 2007).

In many decomposition studies, wood and bark are treated as a single substrate. However, because bark and wood differ strongly in their physical and chemical characteristics, separating their decay processes represents a simple yet informative approach to reduce unexplained variation and to better understand the effects of substrate composition on microbial communities (Jones et al., 2020). Previous studies have demonstrated chemical differences between fungal basidiocarps and fungal decay strategies. However, it remains unresolved whether these chemical patterns primarily reflect intrinsic fungal metabolism or the chemical composition of the substrates being decomposed. This master's thesis aims to investigate the chemical and isotopic dynamics of fungal decomposition of wood and bark by analysing three saprotrophic fungal species and their substrates. The study focuses on differences among biological compartments (basidiocarp, wood, bark), fungal rot types (white rot vs brown rot) and substrate species (hardwood vs softwood). Particular attention is given to organic compounds such as lignin, cellulose, and hemicellulose; lipids, including fatty acids and sterols; and to C and N pathways analysed using stable isotope techniques.

This thesis builds upon previous research on fungal biogeochemistry conducted at the Geolab of the University of Zurich. In particular, it draws on the findings of Grimm (2023), who compared saprotrophic and mycorrhizal basidiocarps and identified significant differences between these lifestyles. Building on these findings, the present study narrows the focus to saprotrophic fungi and incorporates substrate analyses. The overarching aim of this master's thesis is to determine whether saprotrophic fungi are merely a reflection of the substrates they decompose ("what they eat") or whether – and to what extent – they actively modify and are modified by their substrate during wood and bark decomposition. This leads to the central research question:

“How do saprotrophic fungi and their substrates interact during wood and bark decomposition?”

Accordingly, the following main sub-questions arise:

(1) How do fungal, wood and bark compartments differ in chemical composition?
(2) How do different fungal decay strategies (white-rot vs brown-rot) influence the chemical signatures of saprotrophic basidiocarps and their substrates?
(3) How do different substrate types (softwood vs hardwood) influence the chemical signatures of the generalist brown-rot fungus *F. pinicola*?

Based on current literature, I hypothesise that saprotrophic fungi differ significantly in chemical composition from their substrates. Wood and bark are expected to contain higher proportions of structural compounds such as cellulose, hemicellulose and lignin, whereas fungal tissues are expected to show higher concentrations of lipids. I further expect bark and wood to differ chemically, reflecting their different functions within the tree.

Regarding the second research question, differences between rot types are expected in both the substrate and the fungal basidiocarps. Specifically, brown-rotted wood is anticipated to exhibit higher lignin concentrations than white-rotted wood. In contrast, white-rot fungi, which are capable of degrading lignin in their substrates, are expected to contain lignin-derived compounds in their basidiocarps.

Finally, I would expect the brown-rot fungus *F.pinicola* to differ in its decomposition of hardwood and softwood substrates. Previous findings (Xue et al., 2025) show that this fungus preferentially utilises softwood over hardwood substrates, which should be reflected in higher fungal biomass indicators, such as ergosterol and fatty acid concentrations, in softwood-associated samples.

2 Materials and Methods

2.1 Site description

The study area of this thesis is the Laegern forest located within the municipal forests of Wettingen in the canton of Aargau, bordering the canton of Zurich. The Laegern ridge extends over a length of approximately 11 km and a width of 1.5 km and reaches its highest point at 866 m.a.s.l. East and west winds characterise the south-facing slope of the Laegern forest. The forest serves not only as a local recreation area but is also a nature reserve and an important research station. The study site is part of the international CarboEurope IP network, the NABEL (Swiss air quality network) and a long-term forest ecosystem research site of the WSL (Institut für Wald, Schnee & Landschaft) (Swiss FluxNet, 2026). The research station with the measurement tower is situated at 685 m.a.s.l. in a south-facing position, within a montane mixed deciduous forest dominated by beech (*Fagus sylvatica*) and Norway spruce (*Picea abies*) (Wehrli, 2006). The study area was chosen because the flux tower provides valuable scientific data on the C and nutrient cycles of the forest, is close to the University of Zurich, and should not be strongly affected by urban exhaust fumes.

Fungal samples were collected in the Laegern forest near the tower station at two main sites, one above and the other below the road leading to the tower station (Figure 5). The first site has an area of around 200 m^2 and is located on the western side of the flux tower, showing a slight depression or hollow-like shape as the terrain gently slopes downward. The soil can be classified as a calcareous soil, Rendzina. Lime-beech forests mainly dominate the site, but spruce trees are also present. The second site is located on the south-west side of the tower station and consists of a larger area (2300 m^2). The brown earth at this site is slightly more acidic compared to the first site. The main vegetation type is woodruff beech (Wehrli, 2006). Compared to the first site, more saprotrophic fungi growing on softwood trees were found at the second site.

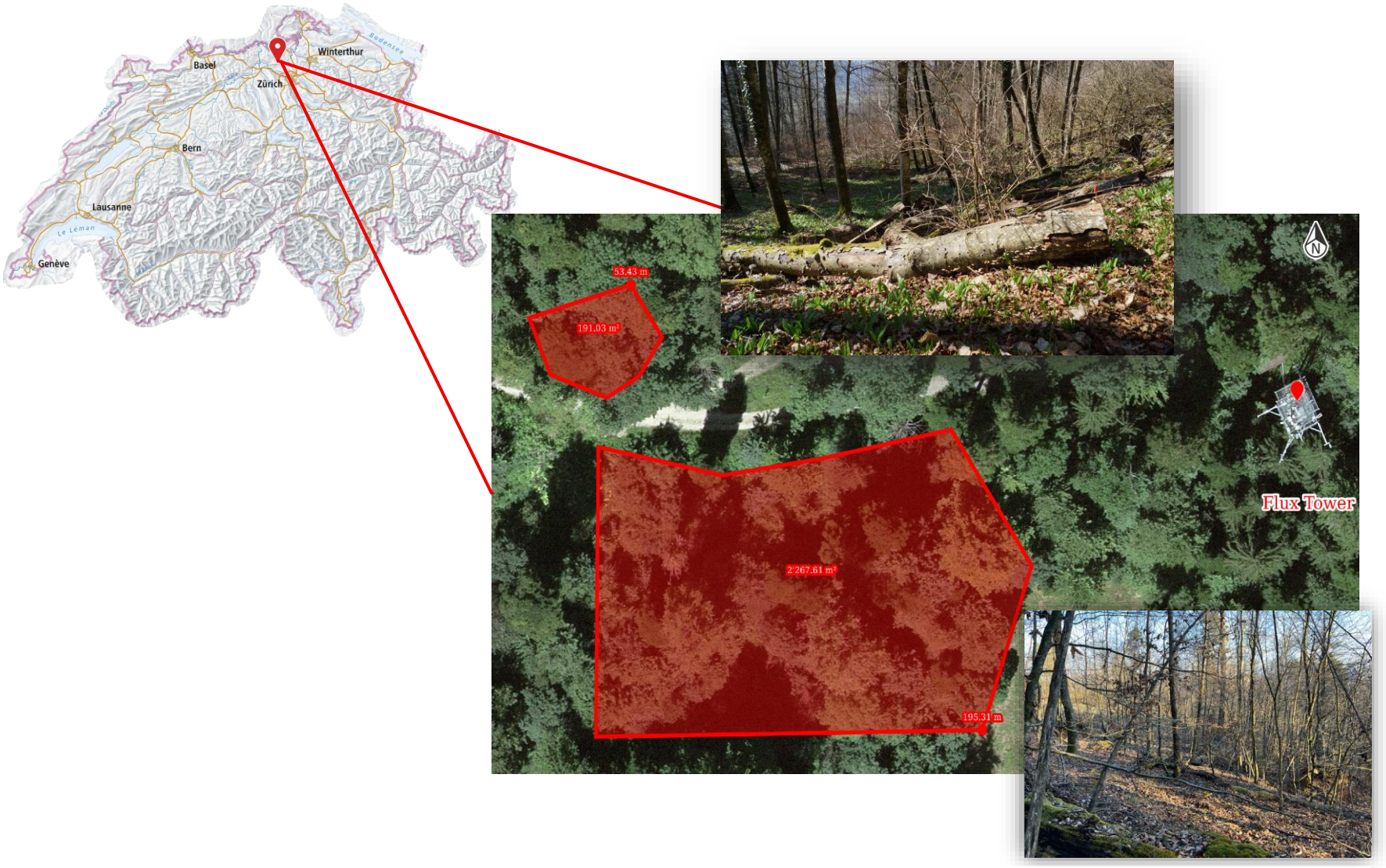

Figure 5: Overview of the study sites in the Laegern forest. Note: Created by the author with Swisstopo.

2.2 Sampling strategy

To familiarise myself with the fungi present in the Laegern forest, I consulted the distribution atlas of SwissFungi WSL, which provides an overview of the abundance, substrates, distribution, phenology and habitat of all known fungal species in Switzerland (Gross et al., 2024). On the first day of fieldwork, the site was surveyed to gain an overview and identify fungal species that might be of interest. In March, most fungi present were wood-decomposing species. On the second day, samples were collected from six fungal species, of which four species were further considered for this thesis (appendix, p.56). The fungi were identified using the book *Holzzersetzende Pilze* (Klug & Lewald-Brudi, 2023). To further reduce the impact of humans and animals on forest roads regarding nutrient input, fungi near paths were not collected. Since wood can exhibit different stages of decomposition, it was essential to collect samples representing a comparable stage of decay to ensure consistency across samples. In my case, the samples are classified as decay class 3 intermediate stage (Garbelotto & Johnson, 2023). Suitable samples were collected by cutting the wood, including the fungal mycelium and basidiocarp, using a hand saw and a circular saw. To prevent contamination, gloves were worn during sampling and the samples were stored in UV-protected loam bags. After labelling the bags, the samples were placed on ice to prevent further degradation.

2.2.1 Brown-rot fungi

F.pinicola is a common bracket fungus in the Basidiomycota. As a generalist, it grows on softwood and hardwood trees in temperate Eurasian forests and is classified as a brown-rot fungus (Xue et al., 2025). As a wound parasite, it requires exposed wound wood for entry and continues to decay after the tree dies. As soon as the tree dies, the lifestyle changes from parasitic to saprotrophic. Young basidiocarps show a white to cream-colored growth margin, while older ones only have a narrow red margin and appear in darker colours (Figure 6) (Klug & Lewald-Brudi, 2023). *F. pinicola* samples were collected from hardwood and softwood trunks and stems in the Laegern forest.

Figure 6: Red-belted bracket (F. pinicola) growing on softwood (left) and hardwood (right). The younger fruiting body shows a brighter orange coloration (right, above). Note: Created by the author.

2.2.2 White-rot fungi

The Tinder fungus (*Fomes fomentarius*), also called hoof fungus due to its large basidiocarps, which are shaped like a horse's hoof, can grow on a large variety of trees, but most typically grows upon hardwoods such as birch and beech trees (Klug & Lewald-Brudi, 2023). It starts growing as a parasite on tree stems and crowns and shifts to saprotrophic nutrition after the tree has died. The tinder fungus can decompose lignin and is therefore classified as a white-rot fungus (Klug & Lewald-Brudi, 2023). *Fomes fomentarius* was collected from hardwood stems in the Laegern forest (Figure 7).

Figure 7: Tinder fungus (Fomes fomentarius) growing on a hardwood stem in the Laegern forest. Note: Created by the author.

The fungal genus *Trametes* belongs as well to the Basidiomycota (Lodi et al., 2025). *Trametes* is a genus of fungi growing in tiled layers in groups or rows on logs and stumps of hardwood and softwood trees. The genus has a widespread distribution and contains about 195 species. This wound parasite is classified as a white-rot fungus. The most studied species is called *Trametes versicolor*. This and another species were found on hardwood trunks and stems in the Laegern forest (Figure 8).

Figure 8: Trametes versicolor growing on a dead hardwood trunk (left). Trametes sp. growing on a hardwood stem (right). Note: Created by the author.

2.3 Sample preparation

To prevent contamination of the samples, the sample preparation procedure was performed in a UV-protected room, either by selecting a room with no windows or by pulling down the shutters. Gloves were worn during the whole working process. Before the samples can be separated into fungal compartments and substrates, they need to be freeze-dried to remove the water in the samples. After each drying period, the samples were weighed to monitor weight loss. If a sample continued to show a decrease in weight compared to the previous measurement, it was subjected to another freeze-drying cycle (1-2 days in the freeze-dryer), indicating that residual water remained. On average, each sample underwent five freeze-drying cycles.

As soon as the samples had been freeze-dried, the fungi and their substrate were separated into their compartments. The basidiocarp and the bark samples could be separated from the wood using a spatula. To separate the wood without mycelia from the wood with mycelia, an electrical saw was used. Sample material containing fungal fruiting bodies of different species on the same substrate or mosses was excluded to prevent undesired influence on measurements. Wood samples include weathered wood, wood without mycelia and wood with mycelia. Wood with mycelia was categorised by defining the so-called zone lines. These are clumps of hard, dark mycelium (Cease et al., 1989). Wood without these zone lines was classified as wood without mycelia. If available, bark samples were also taken from the tree sample. Based on the structure of the basidiocarps, different separation techniques were chosen. While the basidiocarp of *Trametes versicolor* was not further separated into smaller compartments, the basidiocarp of *F. pinicola* was further separated into three anatomical categories: pileipellis (cap surface), trama (structural flesh), and hymenium (fertile layer) (Figure 9). Figure 9 shows an example of a finally prepared sample. The basidiocarps of *Fomes fomentarius* were smaller than those of F. pinicola, which hindered a clear separation of the pileal surface and the trama. Therefore, these were analysed together as structural tissue.

To remove remaining organic particles after sawing, the samples were cleaned using a toothbrush. The samples were then milled using a horizontal mill (Retsch MM 400) at 20 s^{-1} until a homogeneous powder was obtained, which could then be used for laboratory analysis.

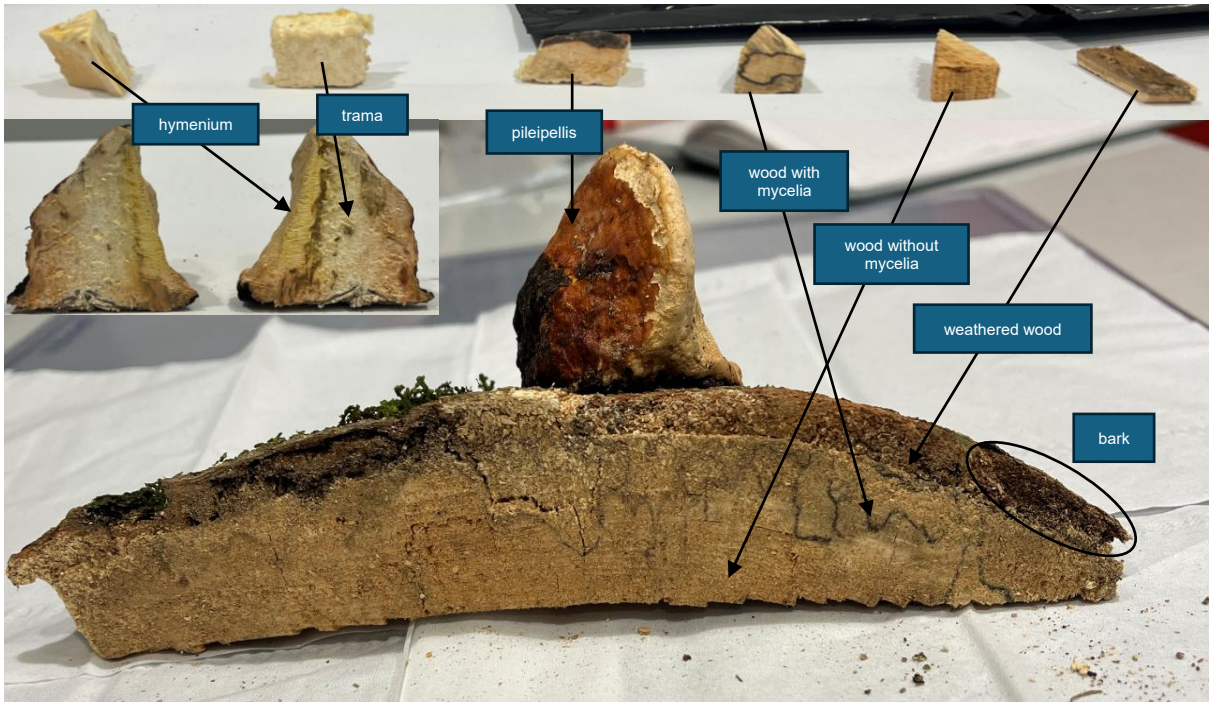

Figure 9: Visualization of the sample separation procedure of Fomitopsis pinicola growing on a softwood tree into bark, weathered wood, wood without mycelium, wood with mycelium, pileipellis, trama and hymenium (from right to left). Note: Created by the author.

2.4 Laboratory analysis

2.4.1 Elemental analysis (EA-IRMS)

Total C (TC) and total N (TN) concentrations, as well as the isotope compositions of ^{13}C and ^{15}N in the fungal, wood, and bark samples, were measured by elemental analysis using the EA-IRMS (elemental analysis – isotope ratio mass spectrometer). TC and TN concentrations include both organic and inorganic elemental composition. Samples are weighted and sealed into tin capsules to be combusted, which breaks them down into gases for isotopic analysis (Gentile et al., 2013). For C values, I weighed 0.8-1.8 mg of the milled sample using an analytical scale. Because N produces weaker signals and has higher background noise in IRMS than C, I weighed twice as much for N measurements (around 2 mg of milled sample). The tin capsules, including the sample, were then placed into the IRMS Delta V PLUS Isotope Ratio MS. Two capsules per sample were measured, and the mean of the two measurements was used for further analysis.

2.4.2 Infrared spectroscopy (DRIFT analysis)

The Invenio R Spectrometer was used to investigate the organic composition of fungal and substrate organic matter. Approximately 2 mg of the milled sample was transferred into Eppendorf vials and dried overnight with potassium bromide (KBr) and a reference sample (Rendzina soil sample) in an oven at 60°C to minimise residual moisture. The presence of water can negatively affect infrared measurements by introducing noise into the spectra. Before the actual measurement, potassium bromide (KBr), an infrared-transparent matrix, was filled into the sample cup and recorded. To acquire an infrared spectrum, both the sample and a reference must be measured, as each spectrum is influenced not only by the sample's absorption properties but also by instrument-specific characteristics. The reference measurement serves to correct for these instrumental effects. After 10 sample measurements, the reference sample was measured again to assess instrument stability and spectral reproducibility. Functional groups have been assigned to the DRIFT spectra according to Ofiti et al. (2021).

2.4.3 Lipid biomarker analysis

For this thesis, the sterols of the low-polar lipid fraction, as well as the fatty acid (FA) fraction, are of particular interest. The whole process follows the approach described by Wiesenberg & Gocke (2015). A more detailed description of the procedures can be found therein (Wiesenberg & Gocke, 2015).

First, an ultrasonic extraction was performed to obtain the total lipid extract (TLE) from the samples. The TLE was further separated into the low polar lipid fraction and the FA fraction. The second separation includes separating the low-polar lipid fraction into an aliphatic C-H fraction, an aromatic C-H fraction, and a heterocompound fraction (NSO). For each sample,

two glass vials are needed: one to record the sample mass and the other for the TLE. First, 150-200 mg of the milled sample was weighed into lockable glass vials. The net sample weight, as well as the weight of the empty second glass vial, was recorded for further calculations. Then, 2 mL of a solvent mixture of dichloromethane (GC grade) and methanol (GC grade) (93:7, v:v) was added to the sample in the glass vial. The vial was closed and shaken using a vortex mixer (Vortex Genie 2). To ensure proper mixing of the sample material with the solvent, the vial containing the sample and solvent was placed in an ultrasonic bath (Brandelin Sonorex) for at least 5 minutes. The samples were then placed in a centrifuge (Megafuge 1.0) for 2 min at 2500 rotations per minute. To transfer the supernatant solvent from the extraction vial to the other vial, a 6 ml glass column equipped with a glass fibre filter on a metal stand must be installed (Figure 10). For each sample, these steps were repeated five times. The dry weight of the lipid extract was determined and named TLE.

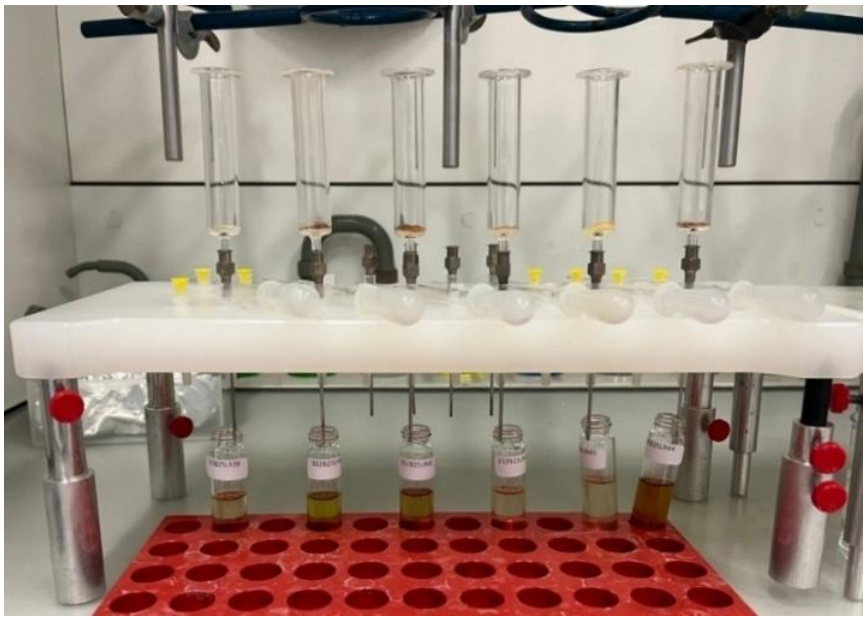

Figure 10: Visualization of the ultrasonic extraction procedure: Installed metal stand containing 6 x a 6 mL glass column and 6 x TLE content. Note: Created by the author.

The TLE fraction was then further separated into low-polarity lipid fractions (N-fraction) and the fatty acid fraction (H-fraction). Previously, 2 glass vials (4 mL each) for each lipid fraction per sample were weighed. For each sample, a 6 mL glass column, a corresponding glass fibre filter, and 1.5-2.0 g of KOH-coated (5%) silica gel 60 were used. For each sample, two round-bottom flasks (50 mL) are used to put in the N-fraction and the H-fraction. For the N-fraction, 30 mL of dichloromethane was used as the solvent; for the H-fraction, 30 mL of DCM: formic acid (99:1) was used as the solvent mixture. After the samples were left in the fume hood overnight to vaporise the solvent, the full vial could be weighed, and the sample mass calculated.

Fatty acids (H-fraction) and alcohols (C-fraction) were analysed using gas chromatography (GC) equipped with a flame ionisation detector (FID) (7890B GC-FID1, Agilent Technologies). Samples were injected into a capillary column using an inert carrier gas as the mobile phase. Compound separation was achieved based on differential interactions with the stationary phase. Quantification was performed using the FID, while compound-specific identification

was confirmed by gas chromatography-mass spectroscopy (GC-MS) through comparison of mass spectra with reference library spectra. Chromatographic data were recorded and processed using Agilent ChemStation software (Agilent Technologies). Retention times were used for compound identification, and peak areas for quantification.
Sterol and fatty acid contents were calculated from chromatographic area-under-the-curve (AUC) values using an internal standard approach. Lipid mass (nmol g^{-1} dry sample) was calculated according to equation (1):

$$Lipid\ mass\ (nmol\ g^{-1}\ dry\ sample) = \frac{F * \left(\frac{area\ lipid}{area\ IS}\right) x\ IS_{added} * R * S}{sample\ weight}$$

where *area lipid* and *area IS* represent the peak areas of the lipid and internal standard, respectively; IS_{added} is the amount of internal standard added; *F* is the response factor; *R* and *S* are conversion factors; and *sample weight* refers to the dry mass of the analysed sample. In this study, no surrogate standard was added, leaving the factor R with 1. The split factor (S) varied by sample between 1, 2 and 4 based on lipid amount in the sample.

The added internal standard (IS_{added}) and the correction factor (F) are calculated as follows:

$$IS\_added\ (nmol) = \frac{V_{IS} * C_{IS} * 1000}{MW_{IS}}$$

where V_{IS} is the volume of IS added (µl), C_{IS} is the concentration of IS solution (mg/ml) and MW_{IS} is the molecular weight of IS (g/mol).

$$F = \frac{N_{IS}}{N_L}$$

Where N_{IS} is the number of C atoms of the internal standard and N_L is the number of C atoms of the respective lipid.

2.5 Statistical analyses

Statistical analyses have been conducted using Python 3 in a Jupyter notebook. The Jupyter notebook environment used was running on ipykernel version 6.4.12. The packages pandas, numpy, matplotlib and seaborn were used for data transformation, visualisation, and numerical calculations. The codes for the analyses can be found in the appendix and are split into model analyses (p. 69-89) and figure visualisation codes, including mean values (p. 89-125).

2.5.1 Linear Mixed Effects Model (LMEM)

A Linear mixed effects model (LMEM) was chosen due to the complex correlation structure and non-independence of the data (Oberg & Mahoney, 2007). To address the three research questions, three separate LMEMs were fitted for the elemental, DRIFT, and lipid biomarker analyses, reflecting the different grouping structures present in the dataset. Compartment, substrate- and rot type effects were treated as fixed effects while the tree ID was used as a random effect. Because multiple samples originated from the same trees, tree ID was modelled as a random intercept to account for non-independence and pseudoreplication. The interaction between compartment and rot type, and between compartment and substrate type, tests whether rot type and substrate type depend on the compartment effect. The three model types have the following structure:

Model A:

$$y_{ik} = \mu + C_i + u_k + \varepsilon_{ik}$$

Model B:

$$y_{ijk} = \mu + C_i + R_j + (CR)_{ij} + u_k + \varepsilon_{ijk}$$

Model C:

$$y_{ijk} = \mu + C_i + S_j + (CS)_{ij} + u_k + \varepsilon_{ijk}$$

C_i = compartment effect
S_j = substrate effect
R_j = rot type effect
u_k = random tree effect
ε = residual error

LMEMs were fitted using the Pymer 4 package in Python, which interfaces with the lme4 and lmerTest packages in R (Jolly, 2018). The model's assumptions included homoscedasticity and residual normality. These were evaluated visually using residual and Q-Q plots (appendix, p. 56). Normality of the data could not always be ensured and was therefore improved by log-transforming the variables where necessary. An overview table including the variables used for the LMEMs and significant results can be found in the appendix (p. 57-66).

2.5.2 Principal component analysis (PCA)

Principal component analysis (PCA) was used to analyse and visualise similarities and differences among compartments, rot types, and substrate types for organic compound groups, fatty acids, and sterols. The PCA reduces the dimensions of the input by replacing a set of variables with a single new variable. Before PCA, all variables were mean-centred and

scaled to unit variance. PCA was therefore performed on the correlation matrix (Webster, 2001). The standard deviation, eigenvalue, proportion of variance and cumulative variance are provided for each PC in the appendix (p. 66 – 69).

2.5.3 Correlation analysis

A correlation analysis was conducted to assess relationships between the sterol ergosterol and fatty acid concentrations. These correlations were performed because ergosterol is widely known as a marker of fungal biomass and may be associated with fungal lipid composition (Högberg, 2006). For the analysis, the fatty acids showing the highest concentrations in fungal basidiocarps ($C_{18:2}$, $C_{18:1}$, $C_{16:1}$) as well as the one only found in fungal basidiocarps ($C_{26:1}$) were chosen. Raw data showed skewed distributions and were therefore log-transformed. Distributional assumptions were assessed using Q-Q plots and residual diagnostics. As deviations from normality and strict linearity remained, the Spearman rank correlation coefficient was used (Schober et al., 2018). Fisher's z transformation was used to test whether correlations differed between hardwood and softwood samples. The results of the correlation analysis are presented in the appendix (p. 56).

3 Results

3.1 Elemental analysis

TC and TN concentrations differed among compartment groups (Figure 11). The LMEM indicated a significant effect of compartment on TC concentrations ($F_{4, 92.56} = 8.63$, $p < 0.001$), with basidiocarps exhibiting significantly lower TC concentrations than all wood compartments ($p < 0.001$, $p = 0.024$) (appendix, p. 60).
In contrast, TN concentrations showed a much stronger compartment effect ($F_{4, 94.10} = 223.45$, $p < 0.001$). Basidiocarps contained significantly higher TN concentrations than all wood compartments ($p < 0.001$), with values approximately 5-13 times higher than those observed in wood samples. Bark also exhibited significantly higher TN concentrations than all wood compartments ($p < 0.001$) (appendix, p. 60). Compared to wood and bark samples, basidiocarp compartment showed considerable variability in TC and TN concentrations (mean ± SD = 44.67 ± 3.61 and 1.16 ± 0.60, respectively).

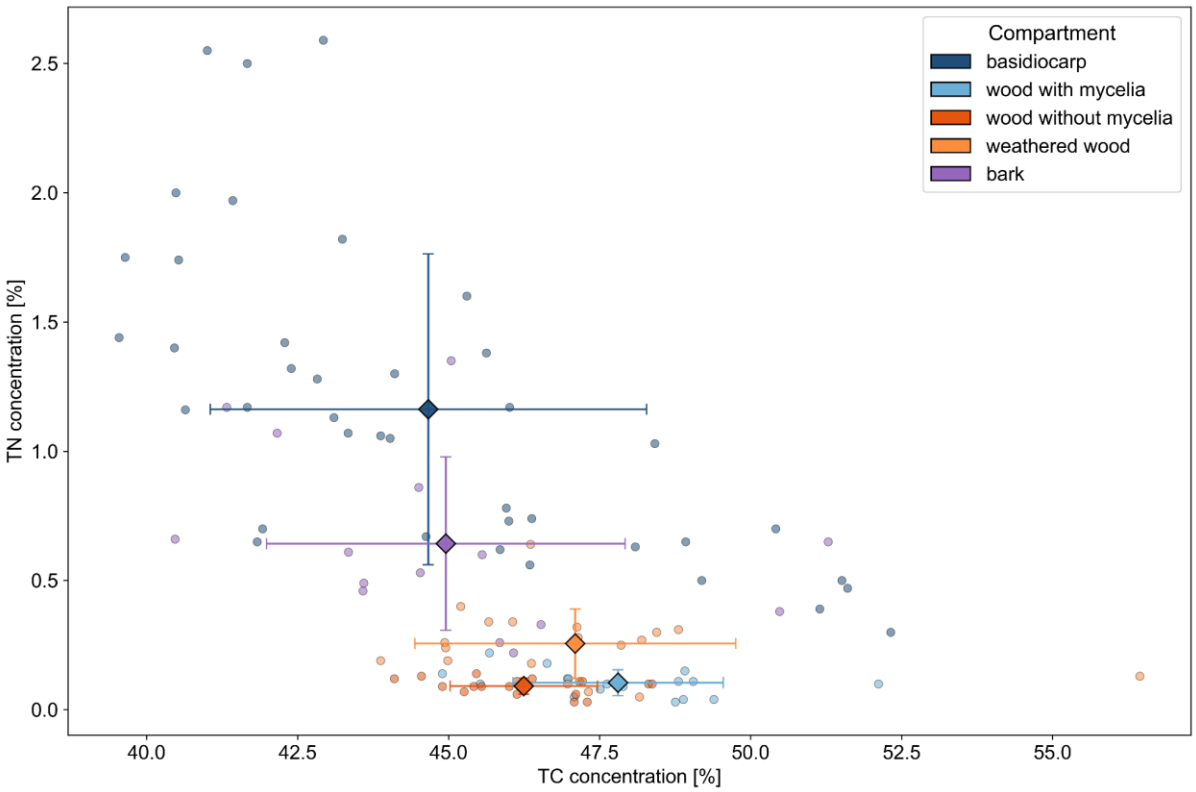

Figure 11: Mean TC and TN concentration [%] (± SD) of the compartments basidiocarp (n=40), wood with mycelia (n=17), wood without mycelia (n=18), weathered wood (n=19) and bark (n=15).

Figure 12 illustrates TC and TN concentrations across compartments, further differentiated by rot type. The results revealed a significant interaction between compartment and rot type for TC concentrations ($F_{4, 92.56} = 8.63$, $p < 0.001$), whereas no such interaction was detected for TN concentrations ($F_{4, 88.29} = 0.64$, $p = 0.63$) (appendix, p. 58).

Specifically, brown-rot basidiocarps exhibited significantly higher TC concentrations than white-rot basidiocarps ($p < 0.001$) and brown-rot bark also showed significantly higher TC concentrations than white-rot bark ($p = 0.05$) (appendix, p. 64).

TN concentrations showed the opposite pattern, with white-rot basidiocarps containing significantly higher TN concentrations than brown-rot basidiocarps ($p = 0.04$). No significant difference in TC or TN concentrations was observed between white-rot and brown-rot wood compartments (weathered wood: $p = 0.18$, wood with mycelia: $p = 0.13$, wood without mycelia: $p = 0.95$).

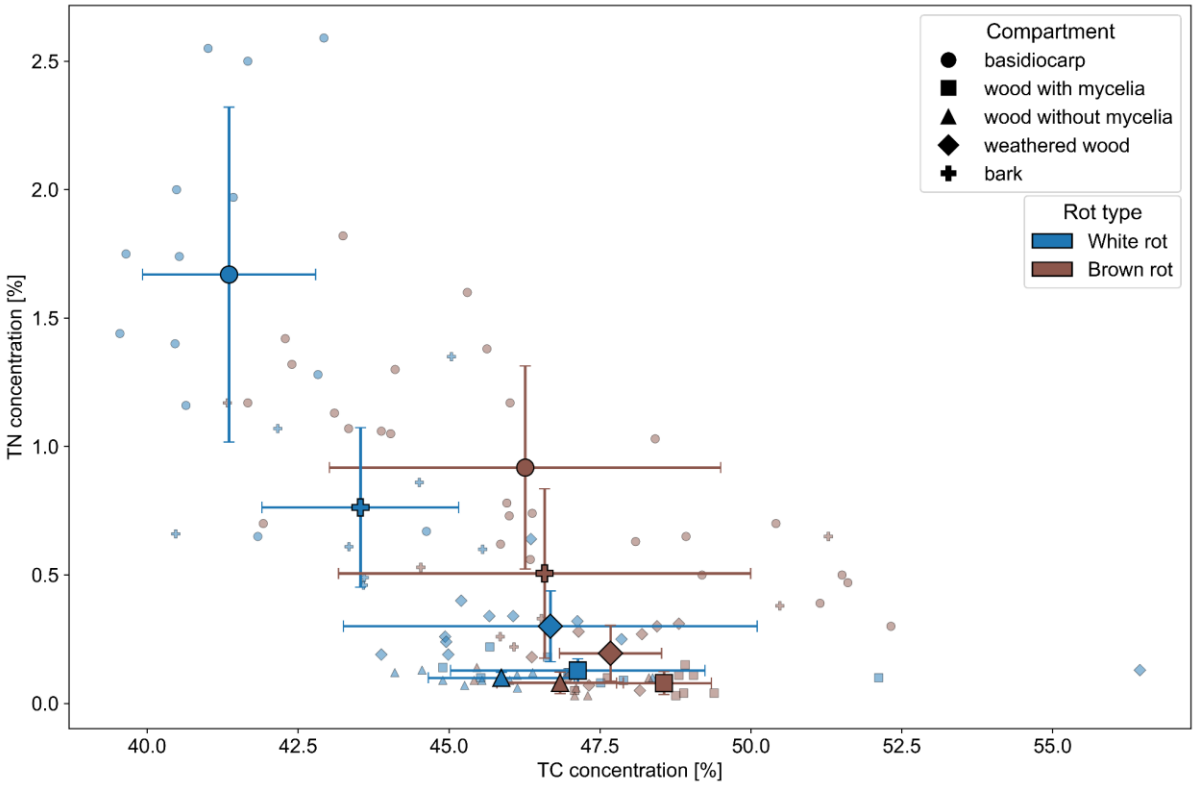

Figure 12: Mean TC and TN concentration [%] (± SD) of white rot and brown rot compartments: basidiocarp (BR: n = 27, WR: n = 13), wood with mycelia (BR: n = 8, WR: n = 9), wood without mycelia (BR: n = 7, WR: n = 11), weathered wood (BR: n = 8, WR: n = 11) and bark (BR: n = 7, WR: n = 8). White rot includes white-rot fungi growing on hardwood substrate, while brown rot includes brown-rot fungi growing on both hardwood and softwood substrates.

TC and TN concentrations of the brown-rot fungus *F. pinicola* were compared between hardwood and softwood substrates (Figure 13). No significant interaction between compartment and substrate was detected for either TC concentrations ($F_{6, 35.77} = 1.43$, $p = 0.23$) or TN concentrations ($F_{6, 36.55} = 1.36$, $p = 0.26$). For TC concentrations, post hoc comparisons nevertheless indicated significantly higher values in the pileipellis and trama when growing on softwood compared to hardwood ($p = 0.025$, $p = 0.009$, respectively).
For TN concentrations, post hoc comparisons revealed significantly higher TN concentrations in hardwood bark, weathered wood and wood without mycelia than the softwood counterparts ($p = 0.02$, $p = 0.02$, $p = 0.04$, respectively) (appendix, p. 65).

Independent of substrate type, the pileipellis exhibited significantly higher TC concentrations than both the trama and the hymenium (HW: $p = 0.007$, $p = 0.009$, SW: $p = 0.03$, $p < 0.001$, respectively). Conversely, the hymenium showed significantly higher TN concentrations compared to the pileipellis when growing on softwood ($p = 0.002$).

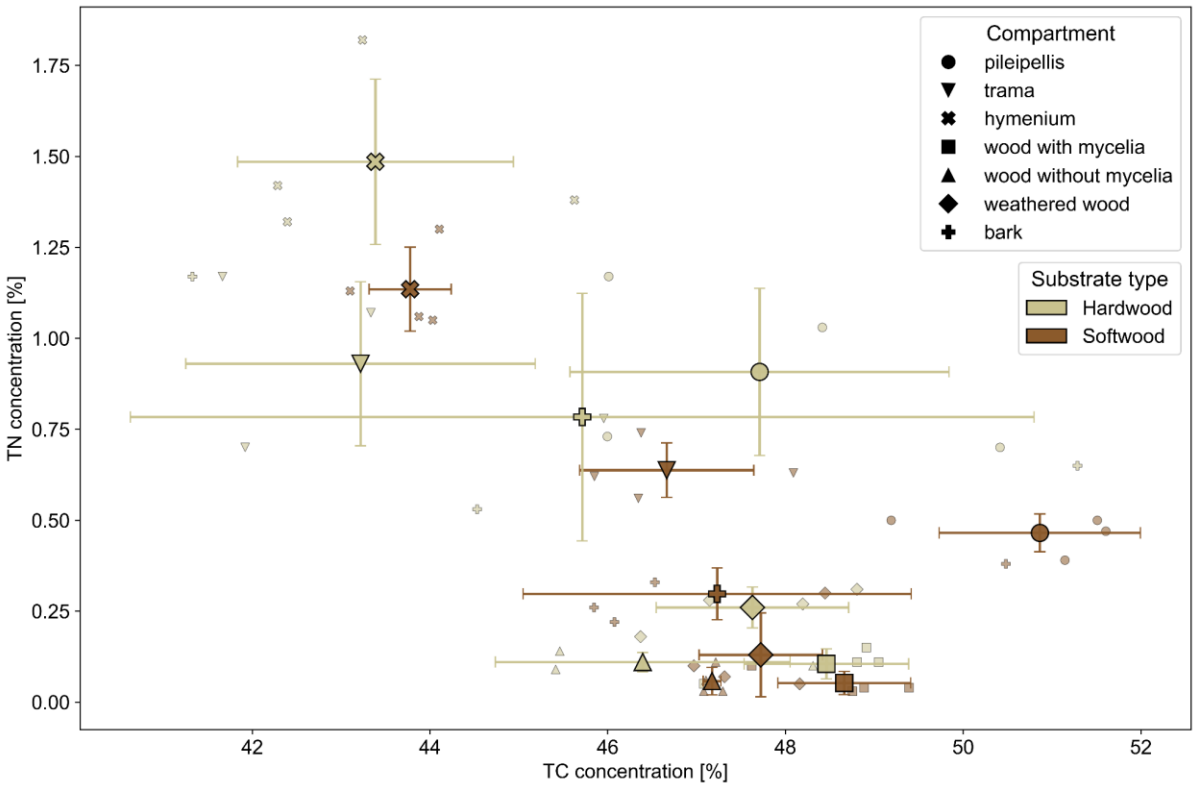

Figure 13: Mean TC and TN concentration [%] (± SD) of brown-rot fungus F. pinicola growing on hardwood vs. softwood including the anatomical features of the basidiocarp (pileipellis (HW & SW: n = 4), trama (HW & SW: n = 4), hymenium (HW & SW: n = 4)), wood with mycelia (HW & SW: n = 4), wood without mycelia (HW: n = 3, SW: n = 4), weathered wood (HW & SW: n = 4) and bark (HW: n = 3, SW: n = 4).

Figure 14 represents isotopic C and N patterns grouped by compartment and visualised for two white-rot fungi growing on hardwood and *F.pinicola* growing on hardwood and softwood. The LMEM revealed a significant effect of compartment on $\delta^{13}C$ concentrations ($F_{4, 93.65} = 55.58$, $p < 0.001$) and $\delta^{15}N$ concentrations ($F_{4, 95.13} = 62.70$, $p < 0.001$). Thereby, basidiocarp samples exhibited significantly higher $\delta^{13}C$ values compared to bark and wood compartments ($p < 0.001$). In contrast, the compartments wood without mycelia and wood with mycelia showed significantly higher $\delta^{15}N$ values compared to weathered wood, bark and the basidiocarp compartments ($p < 0.001$) (appendix, p. 60).

Not for $\delta^{13}C$ but for $\delta^{15}N$, a significant interaction between compartment and rot type was found ($F_{4, 88.26} = 0.28$, $p = 0.89$, $F_{4, 89.92} = 3.32$, $p = 0.01$, respectively). $\delta^{15}N$ values are significantly higher in wood with mycelia compartments of brown-rotted wood compared to white-rotted wood ($p < 0.01$) (appendix, p. 58).

Neither for $\delta^{13}C$ nor for $\delta^{15}N$ values was a significant interaction found between compartment and substrate type ($F_{6, 36.01} = 2.24$, $p = 0.06$, $F_{6, 36.05} = 1.74$, $p = 0.14$, respectively). Post hoc comparisons revealed that the trama growing on hardwood showed significantly higher $\delta^{15}N$ values compared to those growing on softwood ($p = 0.02$) (appendix, p. 65).

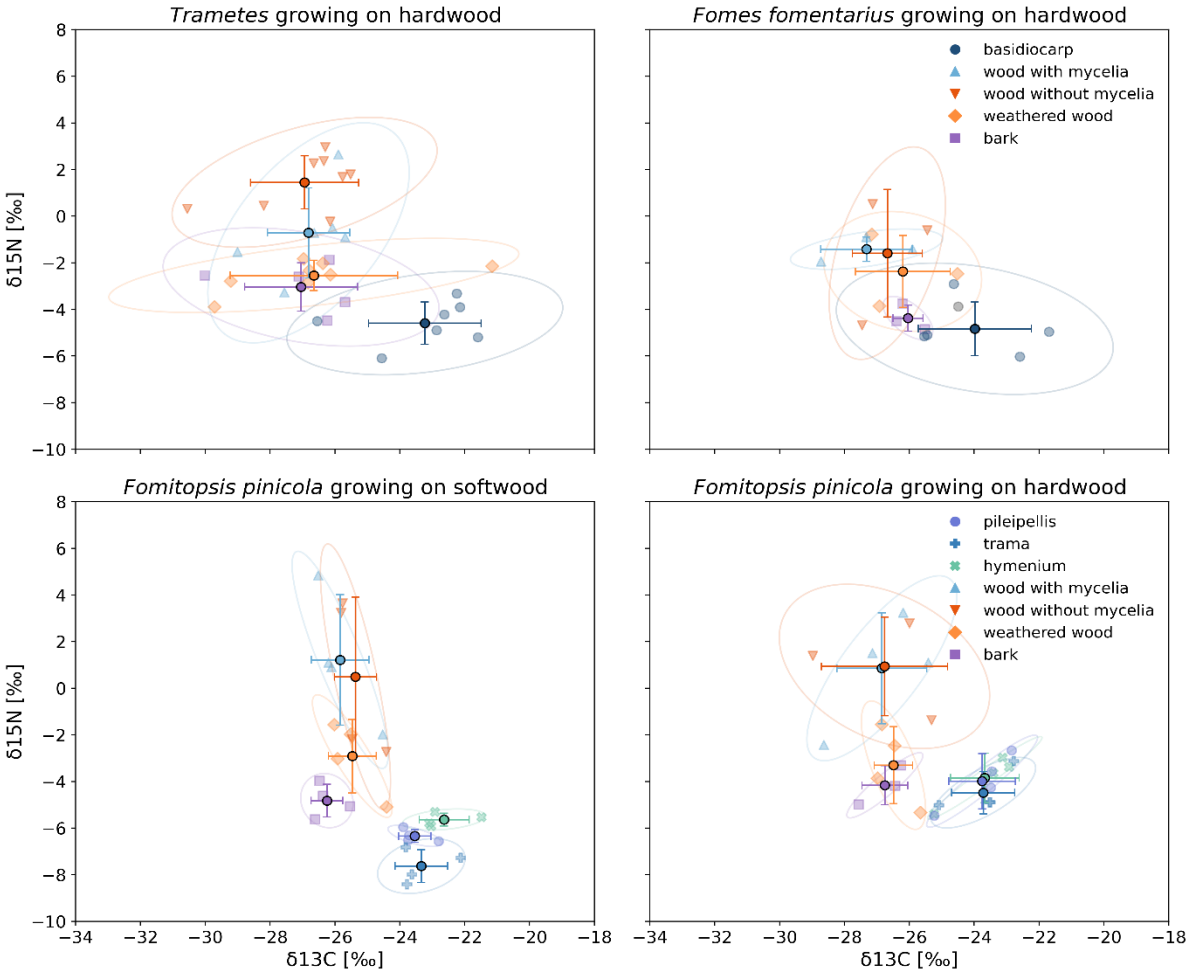

Figure 14: Mean $\delta^{13}C$ and $\delta^{15}N$ [‰] in wood decay fungi (± SD) of two white rot fungal species (Trametes and Fomes fomentarius) and one brown rot fungal specimen (F.pinicola) separated by substrate and categorised by compartments.

3.2 Infrared spectroscopy

The DRIFT spectra show different absorbance intensities in the organic compound composition between compartment groups (Figure 15). All groups indicated a significant compartment effect ($p < 0.001$) (appendix, p. 58-59). Most differences occurred in the fingerprint region (1800 – 1000 cm^{-1}). Pairwise post hoc comparisons revealed that wood compartments contained significantly higher amounts of aliphatic C-H, carboxylic C=O, aromatic C=C (range 1), cellulose, and lignin-like residues compared to bark and basidiocarp samples ($p < 0.001$). Bark also showed significantly higher amounts of lignin-like residues compared to basidiocarp samples ($p < 0.001$) (appendix, p. 61-62).

On the other hand, bark and basidiocarp compartments exhibited significantly higher concentrations of aromatic C=C (range 2) and polysaccharide compounds compared to wood compartments ($p < 0.001$). Aromatic C-H compounds are significantly higher in bark samples compared to wood and basidiocarp compartments ($p < 0.001$).

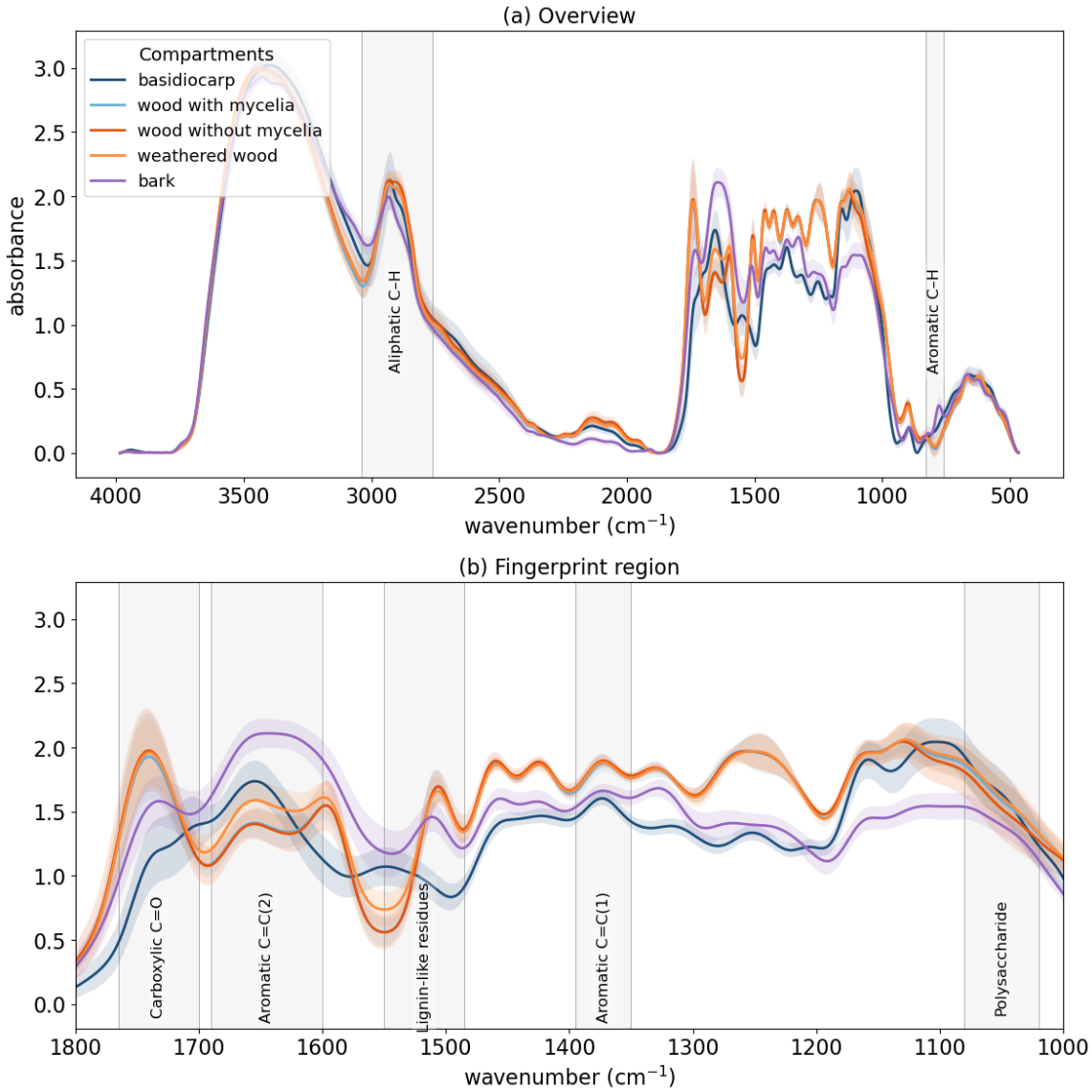

Figure 15: DRIFT spectra (mean ± SD) of wood, fungal and bark compartments overlayed with functional organic groups (Aliphatic C-H, Aromatic C-H, Carboxylic C=O, Aromatic C=C range 1 and range 2, Lignin-like residues, Polysaccharide). a) Overview region (500 – 4000 cm^{-1}), b) fingerprint region (1000 – 1800 cm^{-1}).

Compartment-specific differences in organic compound composition, categorised by rot type, are displayed in Figure 16. Except for polysaccharide, all organic compound groups show a significant interaction between compartment and rot type ($p < 0.001$) (appendix, p. 58-59).
The first two principal components explained 63.5% and 14.4% of the total variation in the DRIFT dataset. Organic compound composition varied strongly with compartment along PC1, with bark and basidiocarp samples clearly separated from wood samples. Bark samples did not cluster according to rot type. Basidiocarp samples clustered according to rot type, with white-rot basidiocarps showing a stronger association with the aromatic C=C range 2.
Wood samples were grouped by rot type and separated along PC2. Brown rotted wood clustered in the direction of lignin, carboxylic and aliphatic functional groups, whereas white rotted wood was associated with cellulose and aromatic C=C range 1. Results of the LMEM further support these patterns: White-rotted wood showed significantly higher values in

aliphatic C-H, carboxylic C=O and cellulose compounds. In contrast, brown-rotted wood had significantly higher levels of lignin-like residues, aromatic C-H, and polysaccharides (appendix, p. 64).

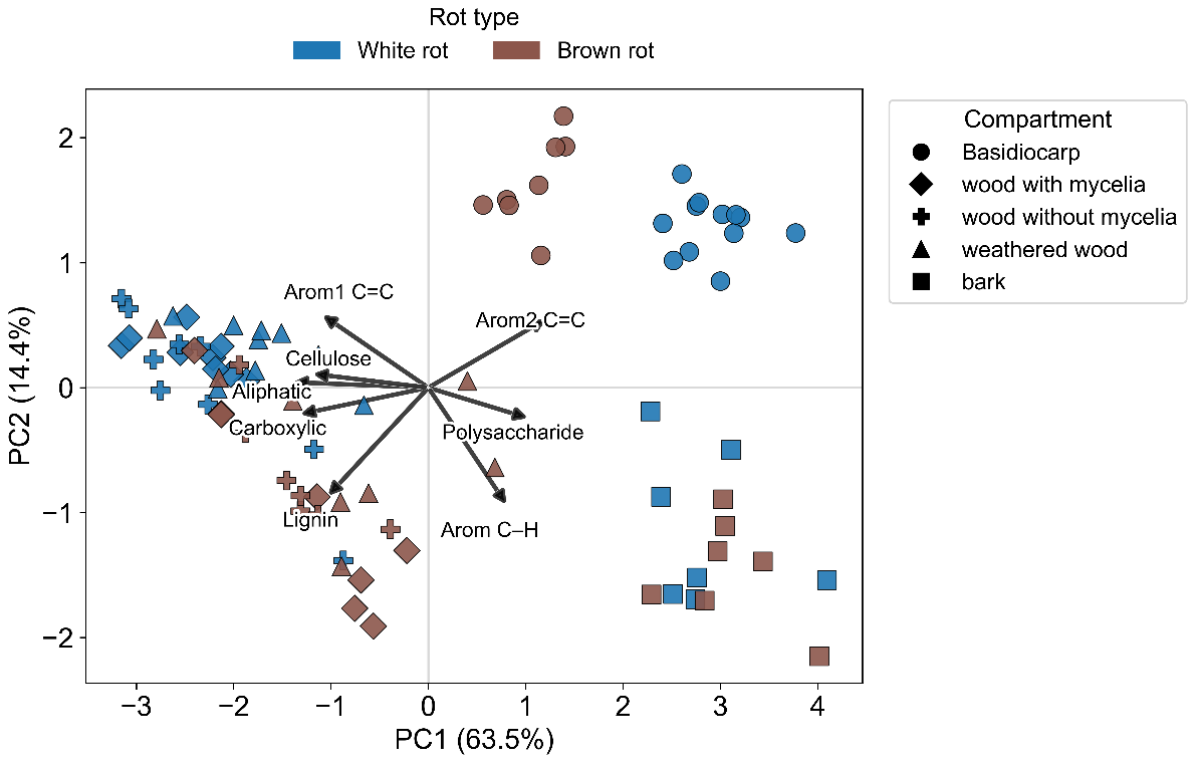

Figure 16: PCA of AUC values from the DRIFT spectra including functional organic groups (Aromatic C=C range 1 and range 2, Aromatic C-H, Polysaccharide, Cellulose, Aliphatic, Carboxylic, Lignin). The panel includes bark, wood (wood with mycelia, wood without mycelia, weathered wood) and basidiocarp samples categorized by rot type. Each point represents the mean PCA score.

In the second PCA plot in Figure 17, basidiocarps of white-rot fungi (*Trametes spp., Fomes fomentarius*) and the brown-rot fungus (*F. pinicola*) are displayed. The first two principal components explained 69.0% and 24.0% of the total variation in the cellulose, aromatic, aliphatic and carboxylic functional groups (Figure 17). Basidiocarps were clearly separated by rot type along PC1, with brown rot samples clustering in the direction of aliphatic, carboxylic, aromatic C=C range 1 and cellulose compounds. White-rot basidiocarps were associated with aromatic C=C (range 2), aromatic C-H, and polysaccharide functional groups, but exhibited substantial within-group variability. The results of the LMEM support the PCA structure: Aliphatic C-H, Carboxylic C=O, aromatic C=C (range 1) and cellulose AUC values are significantly higher in brown-rot basidiocarps compared to white-rot basidiocarps ($p < 0.001$). Aromatic C=C (range 2) AUC values are significantly higher in white-rot basidiocarps compared to brown-rot basidiocarps (appendix, p. 64).

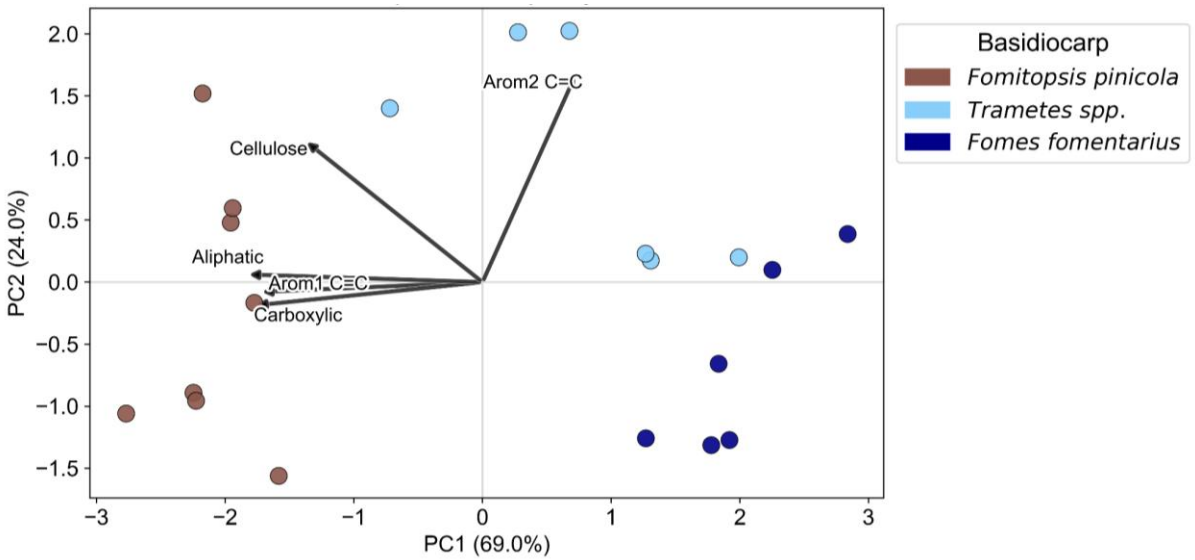

Figure 17: PCA of AUC values from the DRIFT spectra, including functional organic groups (Aromatic C=C range 1 and range 2, Cellulose, Aliphatic C-H, Carboxylic C=O). The panel includes basidiocarp samples categorised by rot type. Brown-rot fungus (F. pinicola (n=8)) and white-rot fungi (Trametes spp. (n=6), Fomes fomentarius (n=6)). Each point represents the mean PCA score.

3.3 Lipid biomarker analysis

3.3.1 Fatty acids

In Figure 18, the ratio of unsaturated to saturated FA and long-chain to short-chain FA is shown. Both ratios indicated a significant compartment effect ($F_{4, 42.74} = 42.75$, $p < 0.001$ and $F_{4, 43.00} = 82.25$, $p < 0.001$, respectively) (appendix, p. 59). Fungal basidiocarp compartments (pileipellis, trama, hymenium) incorporated significantly more unsaturated fatty acids (UFA) to saturated fatty acids (SFA) compared to wood and bark compartments ($p < 0.001$). Wood and bark samples showed the opposite pattern, incorporating significantly more long-chain (LCFA) to short-chain (SCFA) FAs compared to fungal compartments ($p < 0.001$). Bark further incorporated significantly more long-chain than short-chain FAs compared to wood compartments ($p < 0.001$) (appendix, p. 62-63).

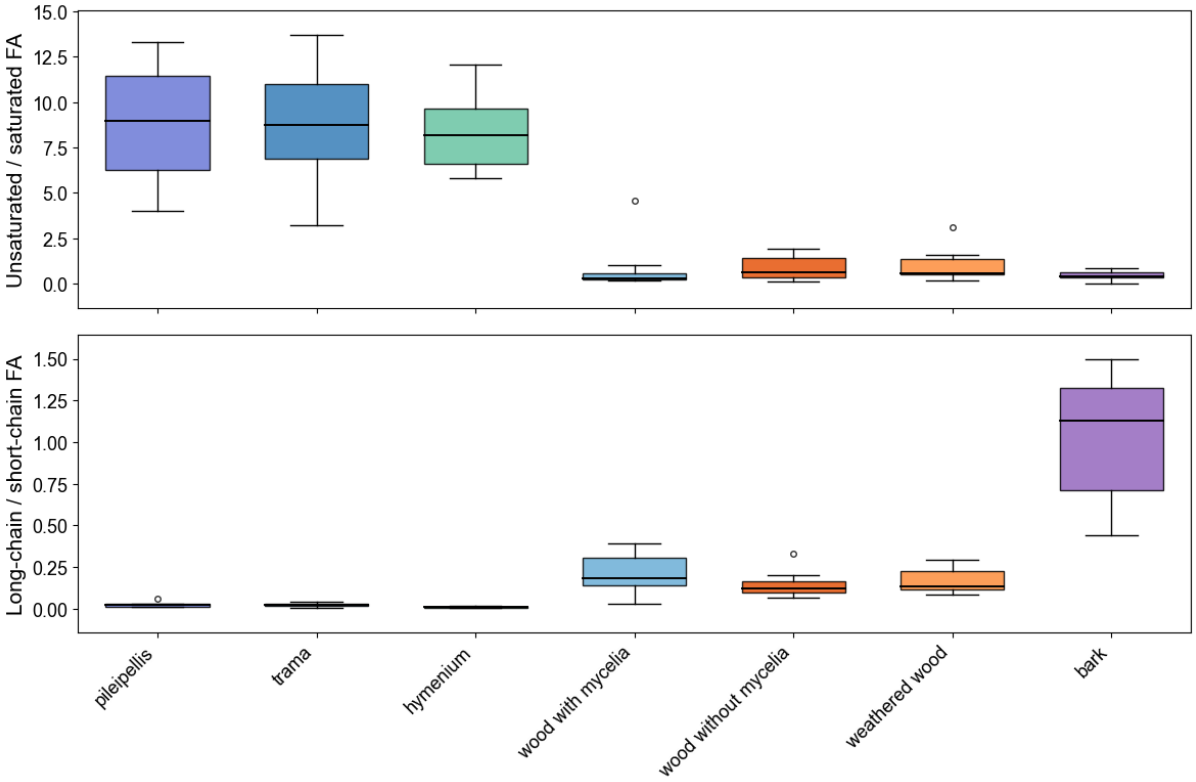

Figure 18: Mean (± SD) of the unsaturated/saturated FA ratio and the long-chain/short-chain FA ratio for each compartment (pileipellis: n = 8, trama: n= 8, hymenium: n = 8, wood with mycelia: n = 8, wood without mycelia: n = 7, weathered wood: n = 8, bark: n = 7). Saturated FA ($C_{14:0}$ – $C_{18:0}$, $C_{20:0}$ - $C_{26:0}$, $C_{28:0}$), unsaturated FA ($C_{15:1}$, $C_{16:1}$, $C_{16:2}$, $C_{18:1}$, $C_{18:2}$, $C_{24:1}$, $C_{25:1}$, $C_{26:1}$), short-chain FA ($C_{14:0}$ - $C_{18:0}$, $C_{15:1}$, $C_{16:1}$, $C_{16:2}$, $C_{18:1}$, $C_{18:2}$), long-chain FA ($C_{20:0}$ – $C_{26:0}$, $C_{28:0}$, $C_{24:1}$, $C_{25:1}$, $C_{26:1}$).

Figure 19 summarises the unsaturated fatty acids (UFA) that were both most abundant in basidiocarps of the brown-rot fungus *F. pinicola* and most strongly differentiated from wood and bark compartments. Of all the unsaturated short-chain fatty acids (USCFA) ($C_{16:1}$, $C_{18:1}$, $C_{18:2}$), the most abundant found in the basidiocarp samples were $C_{18:1}$ and $C_{18:2}$. Unsaturated long-chain fatty acids (ULCFA), for example, $C_{26:1}$, occurred in very small concentrations in the basidiocarps but were not found in wood and bark compartments.

The FAs $C_{16:1}$, $C_{18:1}$, $C_{26:1}$ showed a significant compartment effect ($F_{6, 39.00} = 7.32$, $p < 0.001$, $F_{6, 35.29} = 29.86$, $p < 0.001$, $F_{6, 42.74} = 42.75$, $p < 0.001$, respectively) but $C_{18:2}$ did not ($F_{6, 35.20} = 1.36$, $p = 0.26$). No significant interaction between compartment and substrate type was found ($F_{6, 39.00} = 0.73$, $p = 0.63$, $F_{6, 35.29} = 0.44$, $p = 0.41$, $F_{6, 34.81} = 1.05$, $p = 0.41$, $F_{6, 34.81} = 1.05$, $p = 0.41$, respectively) (appendix, p. 59). The hymenium of *F.pinicola* growing on hardwood showed significantly higher $C_{18:2}$ values compared to those growing on softwood ($p = 0.04$), whereas the pileipellis of *F.pinicola* growing on softwood had significantly higher $C_{26:1}$ ($p = 0.03$) (appendix, p. 66).

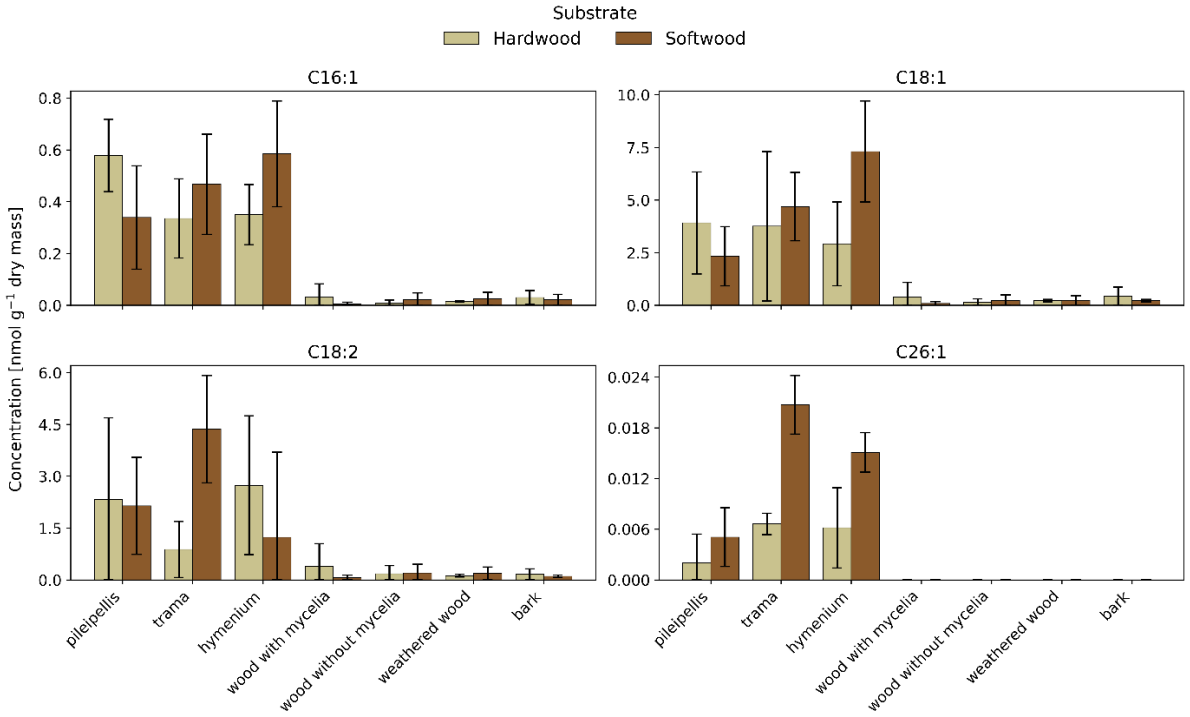

Figure 19: Mean (± SD) of the most abundant FAs found in F. pinicola ($C_{18:1}$, $C_{18:2}$, $C_{16:1}$) and the one found only in basidiocarps ($C_{26:1}$), categorised by substrate. Note that y-axes differ between panels due to differences in concentration ranges among FAs.

3.3.2 Sterols

Overall, in the compartments, five main sterol groups were found: ergosterol, erg. 2 (dehydroergosterol), stigmastanol, β-sitosterol, and ɣ-sitostenone (Table 1). Ergosterol and its derivative erg. 2 showed a significant compartment effect ($F_{4, 43.62}$= 15.67, $p < 0.001$, $F_{4, 41.80}$ = 15.70, $p < 0.001$, respectively) (appendix, p. 59). Compared to its derivative, ergosterol concentrations are 15 to 40 times higher in basidiocarps and around 20 times higher in wood compartments (Table 1). For both, post hoc comparisons revealed that basidiocarp samples had significantly higher values than the wood and bark compartments ($p < 0.001$). Even wood compartments showed significantly higher values compared to bark samples ($p < 0.001$) (appendix, p. 63).

The sterols stigmastanol, β-sitosterol and ɣ-sitostenone were detected exclusively in plant-associated samples (wood and bark compartments) and were absent in fungal basidiocarps. Bark exhibited significantly higher stigmastanol, β-sitosterol and ɣ-sitostenone concentrations compared to wood compartments ($p < 0.001$) (appendix, p. 63).

Table 1: Average sterol concentrations [nmol/g dry mass] (dehydroergosterol, ergosterol, stigmastanol, β-sitosterol, ɣ-sitostenone) ± SD visualised for each compartment. The concentration of the sterols is given in [nmol/g dry mass].

Sterol group	**Compartment**						
	pileipellis	trama	hymenium	wood with mycelia	wood without mycelia	weathered wood	bark
erg. 2 (dehydroergosterol)	**0.084** ± 0.026	**0.187** ± 0.103	**0.096** ± 0.029	**0.061** ± 0.138	**0.018** ± 0.017	**0.041** ± 0.076	**0.001** ± 0.003
ergosterol	**1.288** ± 0.796	**5.160** ± 3.83	**3.645** ± 1.316	**0.262** ± 0.257	**0.339** ± 0.329	**0.258** ± 0.323	**0.062** ± 0.098
stigmastanol	0.000	0.000	0.000	**0.106** ± 0.088	**0.068** ± 0.016	**0.101** ± 0.037	**0.364** ± 0.205
β-sitosterol	0.000	0.000	0.000	**0.331** ± 0.267	**0.235** ± 0.08	**0.479** ± 0.239	**2.288** ± 1.109
ɣ-sitostenone	0.000	0.000	0.000	**0.232** ± 0.191	**0.180** ± 0.219	**0.143** ± 0.102	**0.449** ± 0.24

Figure 20 displays the ergosterol concentration categorised by compartment and substrate type. Ergosterol values indicated not only a significant compartment effect but also a significant interaction between compartment and substrate type ($F_{6,\ 34.73} = 4.20$, $p = 0.002$) (appendix, p. 59). Softwood bark and weathered wood showed significantly higher ergosterol concentrations than their hardwood counterparts ($p = 0.01$, $p = 0.001$) (appendix, p. 66).

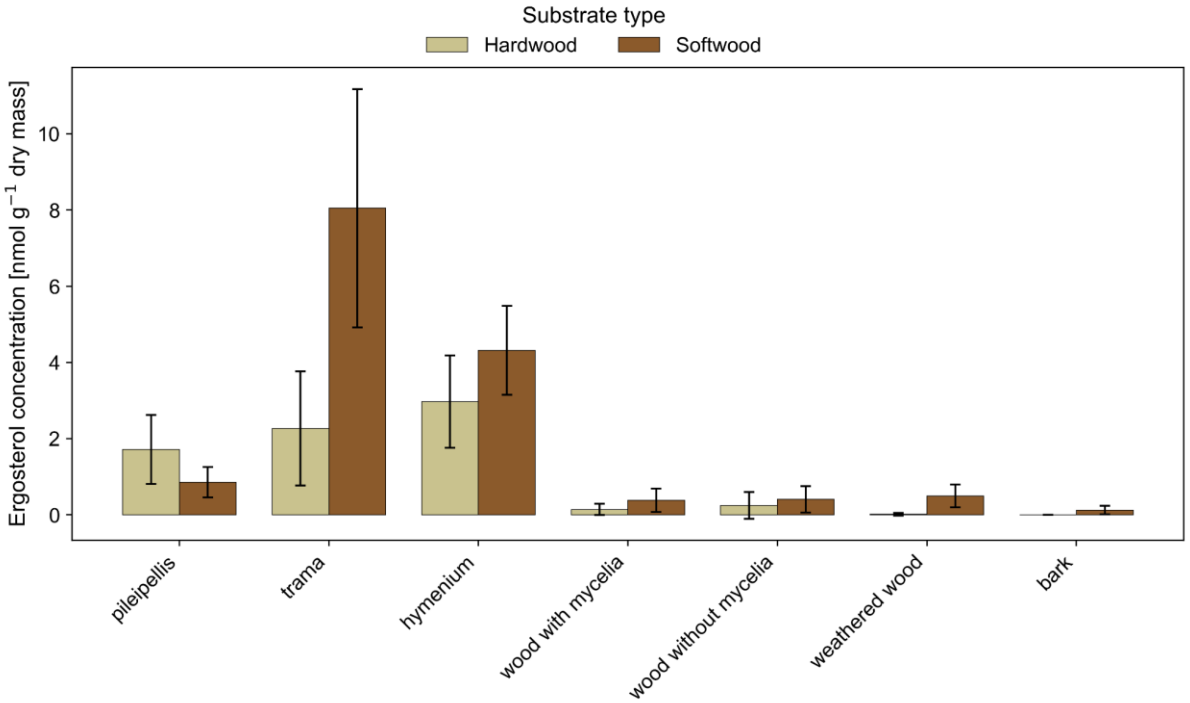

Figure 20: Mean (± SD) ergosterol concentration categorised by compartment and substrate type.

In Figure 21, it was tested whether ergosterol correlates with selected fatty acids ($C_{26:1}$, $C_{18:2}$, $C_{18:1}$, $C_{16:1}$). The Spearman correlation between ergosterol and each fatty acid was classified as strong ($\rho = 0.88$, $\rho = 0.86$, $\rho = 0.82$, $\rho = 0.78$) and significant ($p < 0.001$). Softwood samples showed a slightly stronger correlation compared to hardwood samples, but Fisher's p-value revealed that these differences were not significant ($p > 0.05$) (appendix, p. 56).

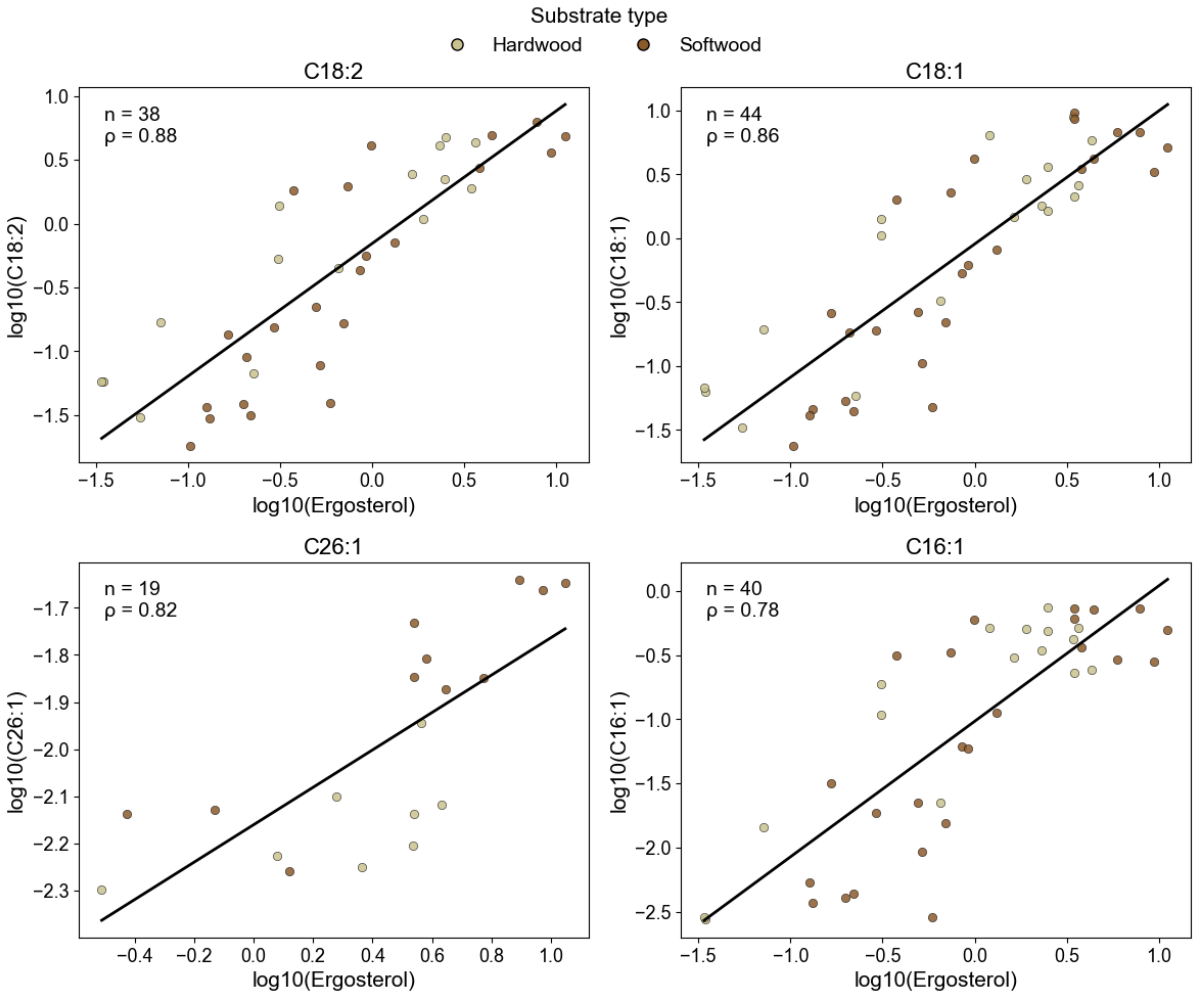

Figure 21: Spearman correlation (ρ) between log-transformed ergosterol and fatty acid ($C_{26:1}$, $C_{18:2}$, $C_{18:1}$, $C_{16:1}$) concentrations. With n = sample size and ρ = Spearman coefficient (rho).

The PCA biplot in Figure 22 provides an overview by including DRIFT and lipid biomarker results for the brown-rot fungus *F. pinicola*, separating compartments by substrate type (hardwood vs softwood). In the first panel, PC1 and PC2 explain 35.2% and 29.7% of the total variance, respectively. Along PC1, fatty acids, sterols and organic compounds vary strongly by compartment, clearly separating bark from basidiocarp samples. Bark samples cluster with the fatty acid groups BCFA, DCA and SLCFA, the sterols stigmastanol, β-sitosterol and ɣ-sitostenone as well as polysaccharide and aromatic C-H compounds, with softwood bark showing a tighter clustering than hardwood bark. BCFA were found to be significantly higher in bark samples than in wood and basidiocarp samples. DCA concentrations are significantly higher in bark compared to basidiocarp samples (appendix, p. 62).

In contrast, basidiocarp samples cluster with the fatty acid groups USCFA, ULCFA, SSCFA, ergosterol and its derivative erg. 2 and aromatic C=C compounds. Along PC2, wood compartments are clearly separated from both basidiocarp and bark samples and are associated exclusively with organic compound groups, including cellulose, lignin-like residues, carboxylic and aliphatic compounds.

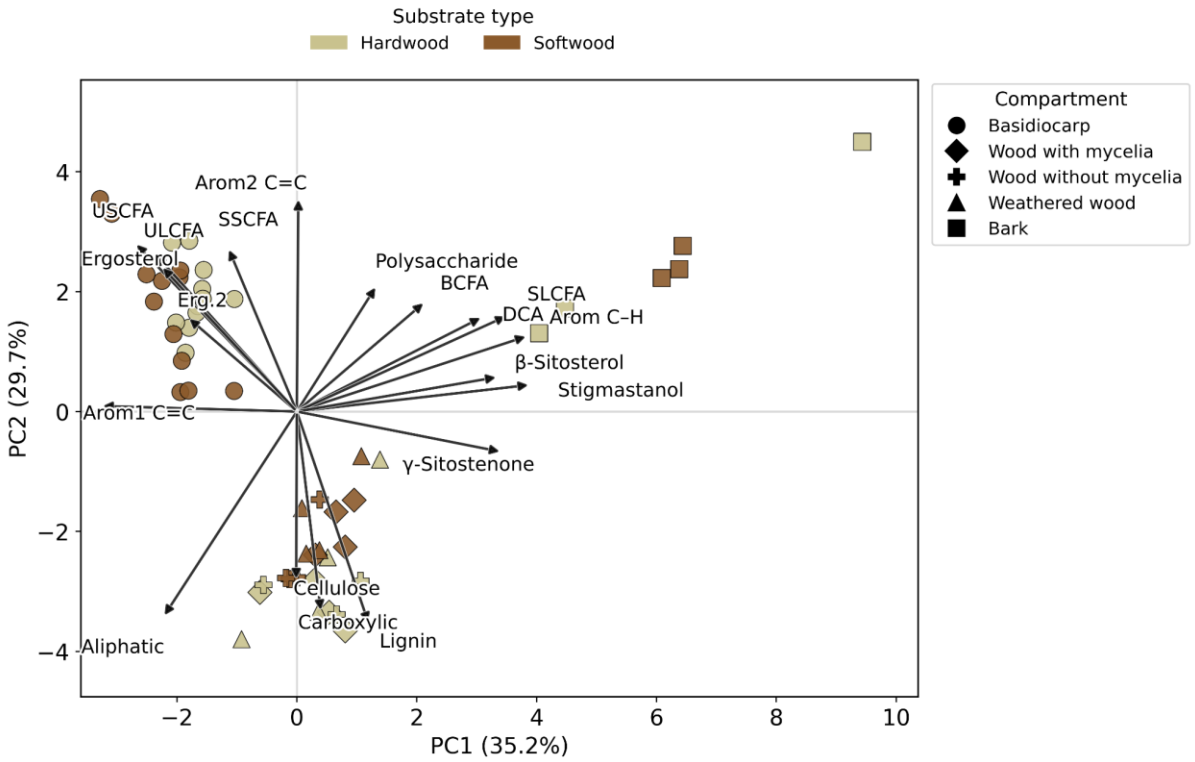

Figure 22: Overview PCA biplot figure including organic compound groups, sterol and fatty acid compounds of the brown-rot fungus F. pinicola.

The second panel illustrates organic compound groups and lipid biomarkers categorised by basidiocarp compartments (pileipellis, trama, hymenium) and substrate type (Figure 24). PC1 and PC2 explain 32.9% and 24.4% of the total variance, respectively. Along PC1, fatty acids, sterols and organic compound groups vary by substrate type, with basidiocarps from softwood trees dominating sterol and FA compounds, as well as carboxylic compounds. Along PC2, pileipellis and hymenium compartments of the basidiocarps are clearly separated, independent of substrate type. Aliphatic compounds dominated the pileipellis compartment and aromatic compounds and SSCFAs the hymenium.

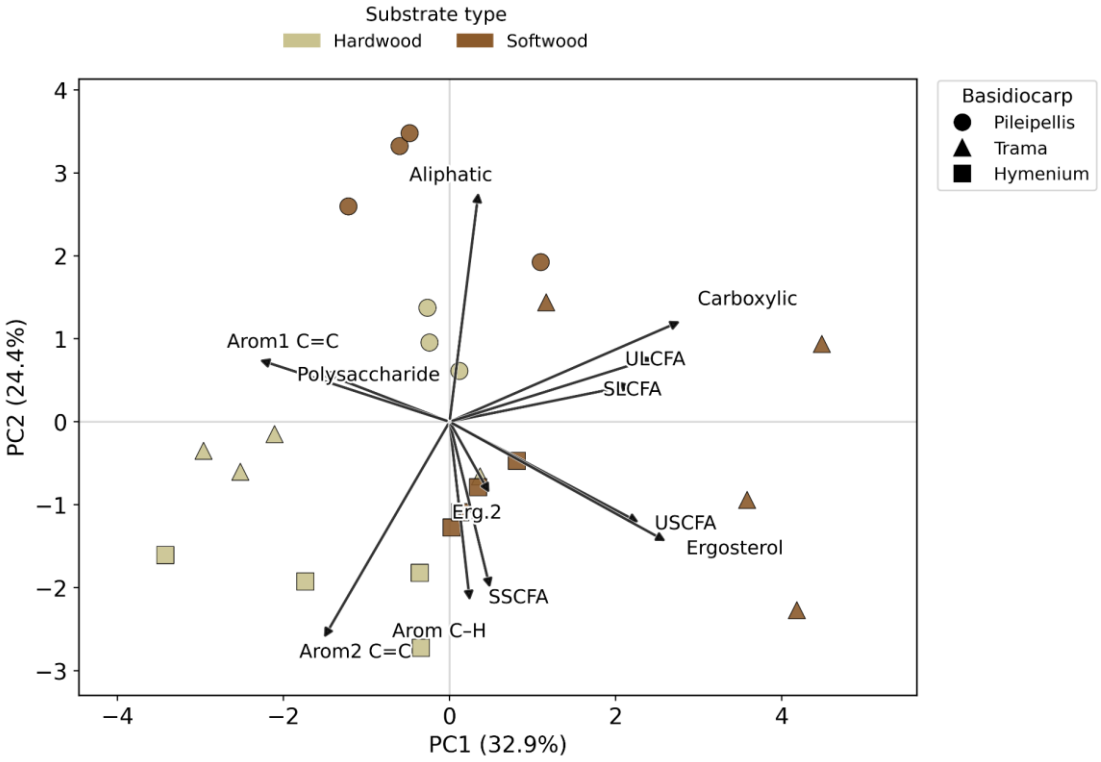

Figure 23: Overview PCA biplot figure including organic compound groups, sterol and fatty acid compounds of the brown-rot fungus F. pinicola.

4 Discussion

4.1 Basidiocarps differ in chemical composition relative to their substrates

TC concentrations were significantly higher in wood compartments (46-47%) than in fungal basidiocarp samples (44%), even though the mean difference was only around 3% (Figure 11). These results align with current literature. Approximately 8% of the global forest C stock is stored in deadwood (Tláskal et al., 2021). Wood and bark comprise approximately 50% C, where most of it is stored in the polysaccharides cellulose and hemicellulose and the biopolymer lignin (Moron et al., 2025). During wood transformation by saprotrophic fungi, most C is released as CO_2 into the atmosphere, while the rest is sequestered within microbial biomass and in soils (Tláskal et al., 2021). Wood-decaying fungi are not able to produce C by themselves and depend on wood as their primary C and energy source (Watkinson et al., 2006). Hobbie et al. (2020) found a range of 40-50% C in saprotrophic fungal basidiocarps, depending on species type. C is stored in fungal cell walls in the form of carbohydrates, proteins, lipids and ashes (Dore et al., 2014). Nonetheless, C is not uniformly distributed within the basidiocarp. Significantly higher values were observed in the fungal pileipellis compared to the fungal hymenium (Figure 13). Direct measurements of elemental C content for individual basidiocarp tissues were not found in the literature. I assume that the C concentration of the pileipellis and hymenium varies due to their distinct structural and functional roles. The pileipellis represents the skin of the basidiocarp that protects the fertile structure (hymenium), reduces water loss and limits gas exchange (Lendzian & Beck, 2021). Due to its exposure to the environment, especially sunlight, it contains different pigments, which are rich in C (Xie et al., 2025). This is expected to be one factor contributing to the higher TC concentrations in pileipellis than in the hymenium. The strong scattering observed in basidiocarp samples may be due to differences among species (Figure 11).

Wood and bark samples (-26.3‰; -26.5‰) exhibited significantly lower $\delta^{13}C$ signature compared to fungal basidiocarps (-23.5‰) (Figure 14), a pattern that has been widely reported in previous studies (Boström et al., 2008; Hobbie et al., 1999, 2020). For example, Gleixner et al. (1993) found a $\delta^{13}C$ enrichment during the formation of fungal chitin from wood cellulose (Gleixner et al., 1993 in Hobbie et al., 1999). Wood has heterogeneous $\delta^{13}C$ values, consisting mainly of ^{13}C-enriched carbohydrates (cellulose and hemicellulose) and ^{13}C-depleted lignin (Benner et al., 1987). Because lignin is more difficult to modify/decompose compared to carbohydrates, it was long assumed that saprotrophic fungi prefer the assimilation of ^{13}C-enriched cellulose and hemicellulose compared to ^{13}C-depleted lignin (Martínez et al., 2011). However, Duran et al. (2024) found that mycelial biomass can be formed from ^{13}C-lignin as the sole source. They cultivated *Agaricus bisporus* on native ^{13}C-lignin and detected ^{13}C labelling in chitin, fatty acids, and ergosterol (Duran et al., 2024). Another factor influencing the higher $\delta^{13}C$ values in fungal basidiocarps may be the loss of ^{13}C-depleted CO_2 during fungal respiration (Boström et al., 2008). Consequently, basidiocarps of wood-decaying fungi are generally enriched in ^{13}C by approximately 3-4‰ compared to their wood substrate

(Boström et al., 2008). Because bark and wood exhibited very similar $\delta^{13}C$ values, $\delta^{13}C$ can not reliably be used to discriminate whether basidiocarp TC concentration originates from wood or bark. The observed enrichment of fungal biomass in ^{13}C relative to their substrates, therefore, likely reflects isotopic fractionation during fungal metabolism rather than differences in source material. The TC and $\delta^{13}C$ results, along with the current literature, indicate that saprotrophic fungi take up C from their substrates.

The DRIFT results revealed strong absorbance in the lignin-like residues and cellulose regions for the wood and bark compartments, with significantly higher values in the wood compartments than in the bark and basidiocarp samples (Figures 15 and 16). These results reflect the dominance of plant-derived structural polymers (lignin) and carbohydrates (cellulose, hemicellulose) in the substrate material and further align with previous TC concentration results (Moron et al., 2025). The low absorbance values observed in basidiocarp samples are consistent with the very low amounts of lignin and cellulose detected in basidiocarp compartments. Chitin-based cell walls provide the structural integrity in basidiocarps (Bowman & Free, 2006). Regarding the difference between wood and bark compartments, Özgenç et al. (2017) stated that bark has a lower carbohydrate content than wood, which supports the results found here. However, when it comes to lignin, the relationship between bark and wood is less straightforward. Bark is generally reported to contain higher lignin concentrations compared to wood (Deb et al., 2021; Özgenc et al., 2017), yet this is not observed by all researchers. Vane et al. (2006) found that bark lignin is more resistant to fungal decay than wood lignin, likely due to the inhibitory effects of extractives such as tannins and suberins. Furthermore, the distinction between outer bark (cork), which is rich in suberin and tannins, and inner bark (phloem), which contains more carbohydrates, may partly explain contradictory findings across studies (Dossa et al., 2018). In contrast, Xiao et al. (2019) analysed eucalyptus wood and bark. They found that lignin content decreased in the order heartwood, sapwood, and bark, suggesting that species-specific differences play an important role.

Aromatic C=C (range 1), aliphatic C-H and carboxylic C=O compounds showed a pattern similar to lignin-like residues and cellulose, with the highest values in wood compartments (Figure 15). Aromatic C=C structures in wood are primarily found within the aromatic ring stretching of lignin but also occur in phenolic extractives (Duran et al., 2024; Ofiti et al., 2021). During wood decomposition, the aromatic structure can be modified, often increasing in relative concentration as cellulose and hemicellulose are broken down (Duran et al., 2024). However, researchers differ in their interpretation of band assignments in DRIFT analysis. Other studies associated this range with aliphatic C-H deformation vibrations in cellulose and hemicellulose, as well as chitin- and melanin-related contributions from fungal tissues (Gupta et al., 2015; Huang et al., 2008; Özgenc et al., 2017; K. K. Pandey & Pitman, 2003).
In wood and bark, aliphatic C-H bonds are present in structural polysaccharides and extractives, which play key roles in protecting trees from environmental stress and inhibiting degradation by fungal saprotrophs (Moron et al., 2025; Karppanen et al., 2007). These extractives mainly include fatty acids, resinous acids, waxes, sterols and some other components such as sugars and proteins (Gao et al., 2024). Furthermore, the peaks observed

around 2918 and 2850 cm^{-1} are characteristic of C-H stretching vibrations and were attributed to aromatic groups in lignin and aliphatic chains of suberin (Md Salim et al., 2021). Carboxylic C=O is attributed to the stretching of free carbonyl groups at around 1739 cm^{-1}. Carbonyl groups occur mainly in the hemicellulose part of wood but also in lignins (Boeriu et al., 2004; H. Chen et al., 2010). Carboxylic acids can be produced through oxidative cleavage of aromatic rings during fungal decomposition (Chen et al., 1983). According to Ofiti et al. (2021) and Boeriu et al. (2004), carboxylic C=O reflects oxygen-rich, oxidised organic matter and is associated with microbial processing. At the same time, carboxyl groups occur in fungal cellular structures, especially in proteins and membrane lipids, as structural components of fatty acids (Gupta et al., 2015). These results confirm that an active wood decomposition process is occurring in the substrate compartments, as further supported by my visual observations of fungal mycelia within the wood and bark substrates.

Aromatic C=C (range 2) and polysaccharide signals were significantly higher in bark and basidiocarp samples compared to wood compartments (Figure 15, Figure 16). The band aromatic C=C (range 2) between 1600 and 1700 cm^{-1} is not specific to aromatic compounds, as it overlaps with the amide I band, a protein-associated compound (Baeva et al., 2020; Gupta et al., 2015; Ofiti et al., 2021; Pandey & Pitman, 2003). For instance, Synytsya et al. (2023) reported that aromatic C=C stretching in triterpenes overlapped with the amide I region, while Vazquez et al. (2000) identified aromatic skeletal vibrations in flavonoids (tannins), which are abundant in bark. One study identified triterpene derivatives and aromatic compounds derived from lignin in *F. pinicola* and *Fomes fomentarius* basidiocarps (Rösecke & König, 2000). These findings indicate that the observed signal may originate from multiple sources, including plant-derived aromatic compounds such as tannins and triterpenes, as well as protein-related components in fungal tissues. In wood and bark samples, polysaccharides mainly occur in the form of cellulose and hemicellulose, but also as starch (Dossa et al., 2018; Moron et al., 2025). Dossa et al. (2018) found that in bark the proportions of galactose, mannose and starch are higher than in wood. Bark tends to show higher concentrations of polysaccharides than wood (Vane et al., 2006). In fungal basidiocarps, polysaccharides such as glucans and chitin are a part of the cell walls (Dore et al., 2014; Synytsya et al., 2023).

All investigated groups of organic compounds were detected across all compartments. Except for aromatic C=C (range 2) and polysaccharide compounds, the organic compound group signals were higher in wood compared to basidiocarps. These findings suggest that the compartments differ significantly in organic compound composition. They also highlight limitations of DRIFT spectroscopy. Overlapping functional groups can lead to ambiguous interpretations and highly band-specific approaches, such as those adopted by Gupta et al. (2015) and Pandey & Pitman (2003). This can lead to an overinterpretation of the data. Therefore, the results should be interpreted with caution, and DRIFT should be regarded primarily as a screening tool.

TN concentrations showed an interesting pattern and a strong differentiation between compartments. Basidiocarp samples exhibited the highest mean concentration (1.16%), followed by bark (0.64‰), weathered wood (0.26%), and the other wood compartments

(0.09% and 0.1%) (Figure 11). It is assumed that fungi preferentially allocate N to sites of growth; for example, N is moved from decayed tree trunks over the fungal mycelia into the basidiocarp (Baeva et al., 2020; Watkinson et al., 2006). Fungi have high N demands due to their protein and chitin synthesis. This demand is even higher in saprotrophic fungi, due to the high metabolic costs of enzyme production, which is required for wood decomposition (Geerits et al., 2025; Trocha et al., 2016). While C is abundant in wood, N availability is often limited and sometimes constrains fungal growth (Barron, 2003; Hobbie et al., 2020). Therefore, wood-decaying fungi may form associations with N_2-fixing bacteria on tree bark to meet their high N demand (Gómez-Brandón et al., 2020). Studies have found that bark generally has two- to tenfold higher N concentrations than wood and serves as a long-term nutrient reservoir (Hadrović et al., 2024; Jones et al., 2020). Compared with the bark-protected wood compartments, weathered wood, fungal basidiocarps, and bark itself were directly exposed to atmospheric N in the forest. Atmospheric N from N oxides (traffic and industry) and NH_3 (agriculture) can deposit on surfaces as wet (rain) and/or dry deposition (particles) (Thimonier et al., 2019). By comparing different Swiss forests with respect to N deposition, a study found that the Laegern forest receives high atmospheric N inputs, especially through dry deposition (particles) (Thimonier et al., 2019). Therefore, the elevated TN concentrations observed in bark, weathered wood and basidiocarps may also reflect the imprint of atmospheric N deposition on these compartments.

Similar to TN concentrations, $\delta^{15}N$ values revealed a strong compartmental differentiation. Significantly lower mean $\delta^{15}N$ values were observed in basidiocarps (-5.25‰), bark (-4.00‰) and weathered wood (-2.75‰) compared to wood with mycelia and wood without mycelia (-0.01‰ and 0.65‰, respectively) (Figure 14). These results indicate that basidiocarps were depleted in ^{15}N relative to their substrates. Comparable basidiocarp $\delta^{15}N$ values were reported by Hobbie et al. (1998), who analysed mycorrhizal and saprotrophic basidiocarps. In a later study, Hobbie et al. (2020) reported average bulk wood $\delta^{15}N$ values of approximately -4‰ in their dataset, although this value may vary by species. This value is much lower than the ones observed for wood in this thesis. Saprotrophic fungi colonising wood preferentially take up the lighter ^{14}N isotope from wood (Barron, 2003; Gómez-Brandón et al., 2020). I presume that as wood decomposition progresses, residual wood becomes enriched in ^{15}N. Weathered wood and bark compartments are further exposed to the environment and may therefore experience enhanced microbial colonisation and N transformation compared to wood protected by bark. Based on the TN and δ15N values, I assume that saprotrophic fungi take up N from their substrates and from other sources, such as bacteria.

The lipid biomarker analysis revealed that total free FA concentrations were significantly higher in basidiocarps than in wood and bark compartments. However, this pattern varied among FA groups. Basidiocarps showed a significantly higher unsaturated-to-saturated FA ratio, whereas wood and bark substrates were characterised by significantly higher long-chain-to-short-chain FA ratios (Figure 18). Saranpää & Nyberg (1987) analysed free FA and sterol composition in wood. They found both SFAs and UFAs in wood, but higher concentrations of UFAs. Studies found that saprotrophic fungi decrease the concentration of free FAs in the substrate during wood decomposition (Gutiérrez et al., 2002; Karppanen et al.,

2008). They accumulate significant amounts of lipids in hyphae and spores, serving as C and energy sources during starvation and spore germination (Reich et al., 2009). They prefer metabolising UFAs, which are more chemically reactive and more readily utilised as substrates than SFAs (Gutiérrez et al., 1999). Especially, USCFAs were abundant in saprotrophic basidiocarps with Oleic acid ($C_{18:1}$) being the main FA, followed by linoleic ($C_{18:2}$) and palmitoleic acid ($C_{16:1}$) (Figure 19). These are known to be the main FAs found in most fungal species (Gutiérrez et al., 2002). Not only in fungal biomass but also in non-infected wood showed linoleic ($C_{18:2}$) and oleic acid ($C_{18:1}$) the highest concentrations compared to other FAs (Saranpää & Nyberg, 1987). These findings suggest that saprotrophic fungi incorporated unsaturated FAs from their substrates.
Compared to wood and basidiocarp compartments, bark showed significantly higher values of SLCFA, DCA and BCFA (Figure 22). Bark extractives such as FAs, alcohols, resins, pigments and tannins make up about 20-40% of bark dry mass (Dossa et al., 2018). These and higher mineral contents make bark a valuable substrate for decomposers. However, terpenoids, resins and polyphenols form a chemical barrier against decomposers, leading to a general slower decay of bark compared to wood substrate (Dossa et al., 2018; Gao et al., 2024; Vane et al., 2006). Vane et al. (2006) found that a relative increase in suberin went along with a decrease in bark cellulose content due to fungal activity. For example, peaks around 2920 cm^{-1} in bark samples are mainly due to the aliphatic chains of suberin (Ferreira et al., 2013 in Md Salim et al., 2021). Suberin is a lipophilic biopolymer found in cork, the outermost layer of bark. It is built among other monomers from SLCFA (with chain lengths $>C_{20}$) and DCAs (Nomberg et al., 2022). BCFAs are a class of FAs characterised by the presence of one or more methyl branches on their C chain. They are considered to be bacterial membrane lipids and not typical plant lipids (Ofiti et al., 2021). I suggest that BCFAs primarily result from bacterial communities living on or in the bark.

For nearly all variables tested, a significant compartment effect was observed. While plant compartments (wood and bark) mainly consist of structural components such as lignin, cellulose, and hemicellulose, fungal basidiocarps showed elevated concentrations of FA and ergosterol. Bark compartments further differed from wood by exhibiting higher levels of extractives, including sterols, BCFA, and DCA. These results support my hypothesis that wood, bark and fungal basidiocarps differ significantly in chemical composition.

4.2 Lipid biomarkers indicate fungal biomass in the substrates

To determine whether wood and bark substrates were infiltrated by saprotrophic fungi, ergosterol concentrations in the samples were analysed. Ergosterol was detected in all compartment groups. The highest concentrations were found in fungal samples, followed by wood samples, with bark showing the lowest ergosterol concentrations (Table 1). Because ergosterol is found exclusively in the membranes of living fungal cells, except for some microalgae, it is frequently used as a fungal biomarker in natural forest ecosystems (Adamczyk et al., 2024; Gessner, 2020; Högberg, 2006). Literature reports that, in addition to ergosterol, certain FAs serve as fungal biomarkers due to their high abundance or exclusive presence in fungal biomass (Gutiérrez et al., 2002). To validate ergosterol as a fungal biomarker, it was

correlated with the most abundant and most exclusive FAs found in *F. Pinicola* ($C_{18:2}$, $C_{18:1}$, $C_{26:1}$, $C_{16:1}$). Ergosterol showed a strong and significant correlation with each FA, independent of substrate type. The strongest correlations were with oleic acid ($C_{18:1}$) and linoleic acid ($C_{18:2}$) (Figure 21). An even higher correlation between ergosterol and linoleic acids ($C_{18:2}$) was found by Högberg (2006). However, it should be noted that Högberg (2006) reported strong correlations between ergosterol and phospholipid-derived fatty acids (PLFA), rather than between ergosterol and free FAs. I expect the correlation to be higher for $C_{18:1}$ and $C_{18:2}$, as these compounds were also detected in the wood and bark compartments, as was ergosterol. On the other hand, $C_{26:1}$ was detected exclusively in basidiocarp samples. Due to the currently limited literature on this compound in fungal tissues, this observation can not be discussed in greater detail here. Further research would be required to evaluate its potential relevance as a biomarker for fungal basidiocarps.

Fungal organisms contain ergosterol as their primary sterol, whereas plant sterols are characterised by a diverse mixture of phytosterols (Dahlin & Ruthes, 2024). Plant sterol groups (stigmastanol, β-sitosterol, and ɣ -sitostenone) were exclusively found in wood and bark compartments, with significantly higher concentrations in bark samples (Table 1). Similar results were reported by Gao et al. (2024), who identified sterols as the primary constituents in bark extracts. According to the literature, wood-decaying fungi decompose both FAs and sterols, although FAs are degraded at a faster rate than sterols (Dorado et al., 2001). The findings of this chapter indicate that fungal biomass was present in all plant compartments (wood and bark), whereas plant-derived biomass may first be modified by fungi before being assimilated.

4.3 Rot type differences are reflected in the substrates

Due to substantial variation in TC concentrations within the basidiocarp compartment, brown-rot and white-rot basidiocarps were analysed separately. Brown-rot basidiocarps showed significantly higher TC concentrations compared to white-rot basidiocarps (46.58%, 43.53%, respectively) (Figure 12). Hobbie et al. (2020) found similar results for the brown-rot fungus *F. pinicola* (46.15%) and the white-rot fungus *Trametes versicolor* (40.44%), both of which are included in this study. However, other white rot species in the same study exhibited higher TC concentrations compared to *F. pinicola*. In their study, TC concentration correlated strongest with lipid content. The lipid concentration in *F.pinicola* was three times higher compared to the concentrations of the white-rot fungus *Trametes versicolor* (Hobbie et al., 2020). I assume that differences in TC concentrations were rather species-specific than rot-type-specific.

All brown-rotted wood compartments showed significantly higher values of lignin-like residue compounds compared to white-rotted wood (Figure 16). On the other hand, white-rotted wood showed significantly higher carboxylic C=O and cellulose values compared to brown-rotted wood (Figure 16). The tendency for brown-rotted wood substrates to align more closely with lignin-like residues and for white-rotted wood substrates to align more closely with cellulose is consistent with known decay mechanisms in the literature (Andlar et al., 2018; Goodell et al., 2008). Brown-rot fungi preferentially and rapidly depolymerise cellulose and hemicellulose,

leaving behind a chemically modified and oxidised lignin residue. On the other hand, white-rot fungi degrade lignin more extensively, reducing the relative dominance of lignin-like signatures in the remaining wood (Castaño et al., 2022; K. Pandey & Pitman, 2003; Xu et al., 2013). For example, the white-rot fungus *Trametes versicolor* prefers to attack lignin rather than hemicellulose and cellulose, thereby producing distinct ligninolytic enzymes such as lignin peroxidase. These catalyse the oxidation of aromatic substrates and modify the lignin structure (Andlar et al., 2018). The role of lignin as a C source for white-rot fungi was long debated. Traditionally, white-rot fungi were viewed as degrading lignin primarily to gain access to polysaccharides (cellulose and hemicellulose), with lignin itself largely mineralised extracellularly and not substantially assimilated into fungal biomass (Hobbie et al., 2020). Recently, Del Cerro et al. (2021) and Duran et al. (2024) found in their studies that white-rot fungi can modify and utilise C from lignin-derived aromatic compounds. They assumed that enzymes extracellularly metabolise lignin into lignin-derived aromatic compounds (syringic acid, 4-HBA, vanillic acid), while some C is metabolically incorporated and other mineralised to CO_2 (Del Cerro et al., 2021). They demonstrated that saprotrophic fungi assimilated and further metabolised lignin as a C source (Duran et al., 2024). However, these mechanisms are still poorly understood and require further research to explain the incorporation of aromatic C into fungal metabolism.

The PCA separated brown-rot basidiocarps clearly from white-rot basidiocarps (Figure 17). Brown-rot basidiocarps exhibited significantly higher aliphatic C-H, carboxylic C=O, aromatic C=C (range 1) and cellulose compounds compared to white-rot basidiocarps. As most literature on brown vs white rot focuses on wood chemistry, it was difficult to find studies discussing groups of organic compounds in basidiocarps. Castaño et al. (2022) compared different decomposing strategies of white-rot and brown-rot fungi and found that at later decay stages, metabolites such as carboxylic acids and aldehydes accumulated more strongly in brown-rot fungi compared to white-rot fungi. In the DRIFT spectra, the band attributed to cellulose may also include contributions of phenolic compounds, which seems more plausible (Ofiti et al., 2021). I assume that the higher aliphatic and carboxylic signals are associated with the elevated TC concentrations observed in brown-rot basidiocarps, likely reflecting fungal lipids and FA-rich compounds. The lipid biomarker analysis supports this interpretation, as FAs were detected in the brown-rot fungus *F. pinicola* (Figures 18 and 19). However, because the lipid biomarker analysis was conducted only for the brown-rot fungus *F. pinicola*, comparisons between brown-rot and white-rot fungi are not possible, which is a limitation of this study.

TN concentrations were significantly higher in white-rot basidiocarps than in brown-rot basidiocarps (Figure 12). Similar patterns were reported previously, with higher TN concentrations in basidiocarps of several white-rot fungi compared to the brown rot species *F.pinicola* (Hobbie et al., 2020). Hobbie et al. (2020) found that higher N concentrations in white-rot basidiocarps correlated with substrate signatures. For example, the white-rot fungus *Trametes versicolor*, a sapwood coloniser, derived a greater proportion of its N from ^{15}N-enriched protein sources, whereas *F.pinicola,* a heartwood coloniser, formed basidiocarps under low N conditions, deriving more N from ^{15}N-depleted chitin (Hobbie et al., 2020). Another

reason may be fungal-bacterial associations that contribute to higher N concentrations in white-rot basidiocarps compared to brown-rot basidiocarps. Jurgensen et al. (1989) observed higher N-fixing bacterial activity in wood colonised by white-rot fungi than in wood colonised by brown-rot fungi. Similar results were found by Tláskal et al. (2021), stating that white-rot fungi are more likely to participate in fungal-bacterial associations (Tláskal et al., 2021). White-rot fungi decay wood faster, partly because bacteria that live with them supply extra N, thereby enabling high enzyme production. In return, white-rot fungi release sugars for bacteria (Jurgensen et al., 1989). Another observation was that all white-rot species were growing on hardwood. Hardwood showed significantly higher TN concentrations compared to softwoods (Figure 13). This may have resulted in higher N uptake and incorporation by white-rot species compared to brown-rot fungi. Furthermore, white-rot fungi rely on enzymes such as laccases and peroxidases, which are proteinaceous and therefore may require a substantial N investment. Given that Hobbie et al. (2020) studied the same fungal species, differences in heartwood vs sapwood colonisation may partly explain the variation in TN concentrations observed between white-rot and brown-rot basidiocarps. This suggests that saprotrophic fungi may access different N sources depending on substrate conditions. In addition, the higher N demand of white-rot fungi is likely linked to their lignin-degrading enzymatic system and associated decomposition strategy.

The results partially support the hypothesis that rot type decay strategies are reflected in both the substrate and the fungal basidiocarp. Brown-rotted wood exhibited significantly higher lignin-like residues than white-rotted wood. Differences in basidiocarp composition were observed between white-rot species and the brown-rot fungus *F.pinicola*. However, it remains unclear whether these reflect rot-type effects or species-specific variation, given that brown-rot fungi were represented by only a single species.

4.4 Limited substrate effects on *F.pinicola* basidiocarp

When comparing fungal basidiocarps, trama and pileipellis of softwood trees showed significantly higher TC concentrations than their counterparts growing on hardwoods (43% and 47%, respectively; 46% and 50%, respectively) (Figure 13). No study was found that analysed C concentrations in *F. pinicola* with respect to substrate type. However, as mentioned in the first chapter, the basidiocarps (especially the pileipellis) contain various pigments that are rich in C. Xie et al. (2025) found that the C concentration varied with basidiocarp colouration. Red varieties showed significantly higher C concentrations compared to yellow varieties. *F.pinicola* is known to produce red to yellowish basidiocarps (Klug & Lewald-Brudi, 2023). However, the colouration depends on the basidiocarps age: it changes from white or reddish-brown when young to dark brown, greyish, or blackish-brown (Klug & Lewald-Brudi, 2023). Furthermore, as they grow, they add new pore layers annually, allowing their age to be determined by counting these layers. In my study, most basidiocarps growing on softwood showed a dark red to nearly black colouration and more pore layers than those growing on hardwoods. I assume the ones on softwood were older than those on hardwood, leading to higher TC concentrations. Between hardwood and softwood substrates, TC concentrations did not differ significantly in either wood or bark (Figure 13). This finding is

unexpected, as wood chemistry generally indicates higher TC concentrations in softwood trees than in hardwoods due to a greater abundance of C-rich compounds such as lignin and terpenes (Lamlom & Savidge, 2003).

Compared to brown-rot basidiocarps growing on softwoods, brown-rot basidiocarps growing on hardwoods showed significantly higher values of aromatic C=C (range 2) and aromatic C=C (range 1). No literature was found comparing aromatic compounds directly in *F. pinicola* growing on hardwood vs softwood. However, Kiseleva et al. (2022) analysed the chemical composition of *F. pinicola* growing on hardwood and softwood substrates. They found that basidiocarps growing on hardwood had significantly higher phenolic content than those growing on softwoods. As aromatic ring structures characterise phenolic compounds, an increase in their abundance would be expected to contribute to stronger aromatic C=C signals (Kiseleva et al., 2022).

Ergosterol and fatty acids showed no significant pattern between hardwood and softwood basidiocarps. However, mean ergosterol concentrations in trama and hymenium were higher in softwood-associated samples, while the ergosterol concentration was higher in pileipellis of hardwoods. Softwood bark and weathered wood samples exhibited significantly higher ergosterol concentrations compared to their hardwood counterparts (Figure 20). Xue et al. (2025) showed that the brown-rot fungus *F. pinicola* prefers softwood substrates over hardwood substrates, reflecting its adaptive evolution in host selection and choice of different ecological niches. I would have expected significantly higher ergosterol values in softwood substrates compared to hardwood substrates, given their preference for softwoods.

While no significant difference in TC concentration was observed between wood and bark compartments of hardwood and softwood trees, TN concentrations differed significantly. TN concentrations were higher in both the wood and bark compartments of hardwood species than in their softwood counterparts (Figure 13). This pattern aligns with previous findings showing higher N concentrations in hardwoods (Hadrović et al., 2024; Hobbie et al., 2020). TN concentrations were also consistently higher in pileipellis, trama and hymenium associated with hardwood substrates compared to softwoods. Despite this consistency, these differences were not statistically significant. However, an interesting pattern across all species is that saprotrophic fungi growing on hardwood tend to show higher TN concentrations than those growing on softwoods. Due to the limited number of species, this observation is only an assumption, but it may be an interesting topic for further research.

Overall, most chemical differences occurred between hardwood and softwood wood substrates, whereas differences between basidiocarp samples were rare. *F. pinicola* samples were collected from both sites in the Laegern forests (sites 1 and 2). Therefore, I assume that basidiocarp age and environmental factors, such as sunlight exposure and moisture, may account for some of the observed differences. Due to the very small differences between hardwood and softwood basidiocarps, the hypothesis that the chemical composition of basidiocarps differed according to their substrate cannot be accepted.

5 Reflection

On the first day of fieldwork, the study site was inspected and a sampling strategy was developed. This strategy aimed to include at least three replicates per species of wood-decaying fungi growing on both hardwood and softwood. On the second day, samples were collected, ensuring that different fungal species were sampled from different trees where possible. One major point of criticism, however, is that some fungal species were collected from the same tree, making it nearly impossible to assign mycelia to their respective basidiocarps. Due to the lack of samples for *F. pinicola* and *Fomes fomentarius*, pseudo-replication could not be entirely avoided. Except for one Trametes species, all samples could be identified to the species level.

In previous experiments conducted at the Geolab, ergosterol could not be detected in fungal samples. In the present study, however, ergosterol was successfully identified. This improvement is likely attributed to changes in sample-handling procedures: the samples were stored in UV-protected bags and kept on ice immediately after collection. According to the literature, ergosterol is remarkably stable when protected from UV light (Gessner, 2020). In addition, the samples were stored and processed in dark rooms to minimise light exposure. These measures appeared to have effectively reduced degradation and prevented the sunlight-induced conversion of ergosterol to vitamin D (Gessner, 2020).

In the lab, samples were visually separated into areas with and without mycelial colonisation based on fungal zones. However, the fungal biomarker ergosterol was detected in all samples, indicating that fungal presence extended beyond visibly colonised regions. This suggests that most of the samples were already fully infiltrated by fungal hyphae. This assumption is further supported by the minimal differences in results observed between wood with mycelia and wood without mycelia. An alternative explanation may be that the substrates chemical signal has dominated that of the mycelia. For future studies, it is recommended to isolate mycelia directly from wood or to sample wood from areas less likely to be colonised, such as regions farther from basidiocarps.

Although a clear and structured sampling strategy was applied, some results must be interpreted with caution due to the low number of replicates, which limited statistical power. In addition, the inclusion of a relatively large number of compartments – up to seven in some cases – provided a broad overview of compartment-specific variation but also increased the datasets complexity. As a result, the discussion focused on the main compartments and only briefly addressed chemical differences within basidiocarp compartments (pileipellis, trama, hymenium). This study should be regarded as a first overview of substrate-related differences in fungal chemistry.

While three different species were included in the white-rot fungi category, only one species represented the brown-rot fungi category. This uneven representation may have contributed to greater intra-group variability among white-rot fungi compared to brown-rot fungi. The substrate-specific results for *F. pinicola* may be interpreted with caution due to the limited

sample size, with only three to four samples analysed per compartment. To obtain more generally valid results, the sample size should be increased in future studies. Due to time constraints, it was not possible to investigate fatty acids and sterol groups in white-rot fungi. Recent studies indicate that white-rot fungi are highly effective at degrading wood extractives, making this a topic worth investigating (Dorado et al., 2001).

6 Conclusion and outlook

The results demonstrated that saprotrophic fungi and their substrates differed significantly in chemical composition. These findings support the first hypothesis, which proposed that wood and bark compartments contain higher proportions of structural compounds such as lignin, cellulose, and hemicellulose. On the other hand, fungal basidiocarps are characterised by higher lipid content. As the results indicated, even the chemical composition of the basidiocarp is heterogeneous. In addition, bark differed significantly from wood, confirming the need to treat these compartments separately. Consistent with its protective function in trees, bark exhibited higher concentrations of lipids (including fatty acids and sterols) as well as N compared to wood samples. Saprotrophic fungi are not just “what they eat” but actively transform substrate compounds into fungal biomass. This was supported by observing plant sterols only in the plant substrate but fungal ergosterol in all compartments. As some of the few organisms capable of degrading lignin, they release simpler organic compounds available to other organisms that contribute to the C cycle of the forest ecosystem. Analysing compartments separately is crucial for understanding their individual contributions to C and N inputs in forest soils. However, to fully analyse nutrient and C transfer processes, future research should aim to trace compound and elemental flows from substrate to fungal tissues. In addition, incorporating different decay stages would help to capture temporal changes in composition and decomposition dynamics.

White-rot and brown-rot fungal basidiocarps, as well as their associated substrates, differed in chemical composition. While the substrate reflects their distinct wood-decay strategies, I assume that differences in basidiocarps are more species-specific. The brown-rot basidiocarps of *F.pinicola* exhibited higher TC concentrations, likely reflecting a greater contribution of lipid compounds. In contrast, white-rot basidiocarps were characterised by higher TN and aromatic C=C (range 2) concentrations, suggesting a higher proportion of protein-related structures. Similarly, substrate composition differed between softwood and hardwood systems: softwood-derived wood and bark contained relatively higher proportions of lignin-like residues, while hardwood samples were comparatively enriched in cellulose. These patterns align with the differing decay mechanisms of brown-rot and white-rot fungi. Future studies may include lipid biomarker analyses of white-rot fungal basidiocarps and a broader range of brown-rot fungal species to enable more direct and robust comparisons. This would allow a more detailed evaluation of potential differences in chemical composition between rot types.

Contrary to my hypothesis, only small differences were observed between *F. pinicola* basidiocarps growing on hardwood and softwood substrates. While the wood substrates differed strongly in organic compound composition and TN concentration, basidiocarps differed only in TC concentration and aromatic compound composition. Future studies should consider the age of basidiocarps, as differences in chemical composition may vary with age. Ergosterol concentrations were significantly higher in softwood wood and bark substrates, but not in the associated basidiocarps. Although not statistically significant, basidiocarps growing on softwood substrates tended to show higher ergosterol concentrations. Based on this result and the known association of brown-rot fungi with softwood substrates, this relationship may be worth further investigation.

7 Acknowledgements

I want to thank Thomy Keller for helping me prepare the fungal and substrate compartments, and Barbara Siegfried and Yves Brügger for guiding me through the experiments in the lab. Furthermore, my thanks go to Binyan Sun and Mike Rowley for answering my questions regarding the statistical analyses. My thanks also go to my family and my boyfriend, Kenny Maurer, for their constant support throughout the year. My special thanks go to my mentor and supervisor, Guido L.B. Wiesenberg, for the assistance and guidance throughout the process of my master's thesis.

8 References

Adamczyk, S., Lehtonen, A., Mäkipää, R., & Camiczyk, B. (2024). A step forward in fungal biomass estimation – a new protocol for more precise measurements of soil ergosterol with liquid chromatography-mass spectrometry and comparison of extraction methods. *New Phytologist*, *241*(6), 2333–2336. https://doi.org/10.1111/nph.19450

Andlar, M., Rezić, T., Marđetko, N., Kracher, D., Ludwig, R., & Šantek, B. (2018). Lignocellulose degradation: An overview of fungi and fungal enzymes involved in lignocellulose degradation. *Engineering in Life Sciences*, *18*(11), 768–778. https://doi.org/10.1002/elsc.201800039

Baeva, E., Bleha, R., Sedliaková, M., Sushytskyi, L., Švec, I., Čopíková, J., Jablonsky, I., Klouček, P., & Synytsya, A. (2020). Evaluation of the Cultivated Mushroom Pleurotus ostreatus Basidiocarps Using Vibration Spectroscopy and Chemometrics. *Applied Sciences*, *10*(22), 8156. https://doi.org/10.3390/app10228156

Baldrian, P. (2017). Microbial activity and the dynamics of ecosystem processes in forest soils. *Current Opinion in Microbiology, 37*, 128–134. https://doi.org/10.1016/j.mib.2017.06.008

Bani, A., Pioli, S., Ventura, M., Panzacchi, P., Borruso, L., Tognetti, R., Tonon, G., & Brusetti, L. (2018). The role of microbial community in the decomposition of leaf litter and deadwood. *Applied Soil Ecology*, *126*, 75–84. https://doi.org/10.1016/j.apsoil.2018.02.017

Barron, G. L. (2003). Predatory fungi, wood decay, and the carbon cycle. *Biodiversity*, *4*(1), 3–9. https://doi.org/10.1080/14888386.2003.9712621

Benner, R., Fogel, M. L., Sprague, E. K., & Hodson, R. E. (1987). Depletion of ^{13}C in lignin and its implications for stable carbon isotope studies. *Nature*, *329*(6141), 708–710. https://doi.org/10.1038/329708a0

Blanchette, R. A., Otjen, L., Effland, M. J., & Eslyn, W. E. (1985). Changes in structural and chemical components of wood delignified by fungi. *Wood Science and Technology*, *19*(1), 35–46. https://doi.org/10.1007/BF00354751

Boeriu, C. G., Bravo, D., Gosselink, R. J. A., & van Dam, J. E. G. (2004). Characterisation of structure-dependent functional properties of lignin with infrared spectroscopy. *Industrial Crops and Products*, *20*(2), 205–218. https://doi.org/10.1016/j.indcrop.2004.04.022

Boström, B., Comstedt, D., & Ekblad, A. (2008). Can isotopic fractionation during respiration explain the ^{13}C-enriched sporocarps of ectomycorrhizal and saprotrophic fungi? *New Phytologist*, *177*(4), 1012–1019. https://doi.org/10.1111/j.1469-8137.2007.02332.x

Bowman, S. M., & Free, S. J. (2006). The structure and synthesis of the fungal cell wall. *BioEssays*, *28*(8), 799–808. https://doi.org/10.1002/bies.20441

Camilleri, E., Blundell, R., Baral, B., Karpiński, T. M., Aruci, E., & Atrooz, O. M. (2024). A comprehensive review on the health benefits, phytochemicals, and enzymatic constituents for potential therapeutic and industrial applications of *Turkey tail* mushrooms. *Discover Applied Sciences*, *6*(5), 257. https://doi.org/10.1007/s42452-024-05936-9

Castaño, J. D., Muñoz-Muñoz, N., Kim, Y. M., Liu, J., Yang, L., & Schilling, J. S. (2022). Metabolomics Highlights Different Life History Strategies of White and Brown Rot Wood-Degrading Fungi. *mSphere*, *7*(6), e00545-22. https://doi.org/10.1128/msphere.00545-22

Cease, K. R., Blanchette, R. A., & Highley, T. L. (1989). Interactions between *Scytalidium* species and brown- or white-rot basidiomycetes in birch wood. *Wood Science and Technology*, *23*(2), 151–161. https://doi.org/10.1007/BF00350937

Chen, C.-L., Chang, H.-M., & Kirk, T. K. (1983). Carboxylic Acids Produced Through Oxidative Cleavage of Aromatic Rings During Degradation of Lignin in Spruce Wood by Phanerochaete Chrysosporium. *Journal of Wood Chemistry and Technology*, *3*(1), 35–57. https://doi.org/10.1080/02773818308085150

Chen, H., Ferrari, C., Angiuli, M., Yao, J., Raspi, C., & Bramanti, E. (2010). Qualitative and quantitative analysis of wood samples by Fourier transform infrared spectroscopy and multivariate analysis. *Carbohydrate Polymers*, *82*(3), 772–778. https://doi.org/10.1016/j.carbpol.2010.05.052

Swiss FluxNet. (2026, February 13). *CH-LAE, Laegeren*. https://www.swissfluxnet.ethz.ch/index.php/sites/site-info-ch-lae/

Dahlin, P., & Ruthes, A. C. (2024). Loss of Sterol Biosynthesis in Economically Important Plant Pests and Pathogens: A Review of a Potential Target for Pest Control. *Biomolecules*, *14*(11), 1435. https://doi.org/10.3390/biom14111435

Deb, D., Roy, P., & Erickson, R. (2021). Wood and bark lignin contents of trees from deciduous forests of eastern India. *Experimental Results*, *2*, e28. https://doi.org/10.1017/exp.2021.18

Del Cerro, C., Erickson, E., Dong, T., Wong, A. R., Eder, E. K., Purvine, S. O., Mitchell, H. D., Weitz, K. K., Markillie, L. M., Burnet, M. C., Hoyt, D. W., Chu, R. K., Cheng, J.-F., Ramirez, K. J., Katahira, R., Xiong, W., Himmel, M. E., Subramanian, V., Linger, J. G., & Salvachúa, D. (2021). Intracellular pathways for lignin catabolism in white-rot fungi. *Proceedings of the National Academy of Sciences*, *118*(9), e2017381118. https://doi.org/10.1073/pnas.2017381118

Dorado, J., Van Beek, T. A., Claassen, F. W., & Sierra-Alvarez, R. (2001). Degradation of lipophilic wood extractive constituents in *Pinus sylvestris* by the white-rot fungi *Bjerkandera sp*. and *Trametes versicolor*. *Wood Science and Technology*, *35*(1), 117–125. https://doi.org/10.1007/s002260000077

Dore, C. M. P. G., Alves, M. G. das C. F., Santos, M. da G. L., de Souza, L. A. R., Baseia, I. G., & Leite, E. L. (2014). Antioxidant and anti-inflammatory properties of an extract rich in Polysaccharides of the mushroom *Polyporus dermoporus*. *Antioxidants, 3*(4), 730–744. https://doi.org/10.3390/antiox3040730

Dossa, G. G. O., Schaefer, D., Zhang, J.-L., Tao, J.-P., Cao, K.-F., Corlett, R. T., Cunningham, A. B., Xu, J.-C., Cornelissen, J. H. C., & Harrison, R. D. (2018). The cover uncovered: Bark control over wood decomposition. *Journal of Ecology*, *106*(6), 2147–2160. https://doi.org/10.1111/1365-2745.12976

Duran, K., Kohlstedt, M., Van Erven, G., Klostermann, C. E., America, A. H. P., Bakx, E., Baars, J. J. P., Gorissen, A., de Visser, R., de Vries, R. P., Wittmann, C., Comans, R. N. J., Kuyper, T. W., & Kabel, M. A. (2024). From ^{13}C-lignin to ^{13}C-mycelium: *Agaricus bisporus* uses polymeric lignin as a carbon source. *Science Advances*, *10*(16), eadl3419. https://doi.org/10.1126/sciadv.adl3419

Eichlerová, I., Homolka, L., Žifčáková, L., Lisá, L., Dobiášová, P., & Baldrian, P. (2015). Enzymatic systems involved in decomposition reflects the ecology and taxonomy of saprotrophic fungi. *Fungal Ecology*, *13*, 10–22. https://doi.org/10.1016/j.funeco.2014.08.002

Gadd, G. M. (2013). Geomycology: Fungi as agents of biogeochemical change. *Biology and Environment: Proceedings of the Royal Irish Academy*, *113B*(2), 139–153.

Gadd, G. M., Dyer, P. S., & Watkinson, S. C. (Eds.). (2007). *Fungi in the environment*. Cambridge University Press.

Gao, C., Cui, X., & Matsumura, J. (2024). Multidimensional exploration of wood extractives: A review of compositional analysis, decay resistance, light stability, and staining applications. *Forests*, *15*(10), 1782. https://doi.org/10.3390/f15101782

Garbelotto, M., & Johnson, M. G. (2023). High-throughput DNA metabarcoding of stem sections from trees with cavities describes fungal communities associated with variable wood decay, position on stem and tree species. *Forests*, *14*(6), 1070. https://doi.org/10.3390/f14061070

Geerits, C. H. A., Haar, M. K., Knobel, K. C. C., Vincken, J.-P., & Kabel, M. A. (2025). A comprehensive approach to the chemical analysis of fungal biomass – the pitfalls of nutritional standardization. *Food Chemistry*, *492*, 145443. https://doi.org/10.1016/j.foodchem.2025.145443

Gentile, N., Rossi, M. J., Delémont, O., & Siegwolf, R. T. W. (2013). δ^{15}N measurement of organic and inorganic substances by EA-IRMS: A speciation-dependent procedure. *Analytical and Bioanalytical Chemistry*, *405*(1), 159–176. https://doi.org/10.1007/s00216-012-6471-z

Gessner, M. O. (2020). Ergosterol as a measure of fungal biomass. In F. Bärlocher, M. O. Gessner, & M. A. S. Graça (Eds), *Methods to Study Litter Decomposition: A Practical Guide* (pp. 247–255). Springer International Publishing. https://doi.org/10.1007/978-3-030-30515-4_27

Gómez-Brandón, M., Probst, M., Siles, J. A., Peintner, U., Bardelli, T., Egli, M., Insam, H., & Ascher-Jenull, J. (2020). Fungal communities and their association with nitrogen-fixing bacteria affect early decomposition of Norway spruce deadwood. *Scientific Reports*, *10*(1), 8025. https://doi.org/10.1038/s41598-020-64808-5

Goodell, B., Qian, Y., & Jellison, J. (2008). Fungal decay of wood: Soft Rot—Brown Rot—White Rot. In *Development of Commercial Wood Preservatives* (Vol. 982, pp. 9–31). American Chemical Society. https://doi.org/10.1021/bk-2008-0982.ch002

Grinhut, T., Hadar, Y., & Chen, Y. (2007). Degradation and transformation of humic substances by saprotrophic fungi: Processes and mechanisms. *Fungal Biology Reviews*, *21*(4), 179–189. https://doi.org/10.1016/j.fbr.2007.09.003

Gross, A., Blaser, S., & Senn-Irlet, B. J. (2024). *Swiss Fungi: National data and information centre of Swiss fungi* (Version 2) [Data set]. Swiss Federal Institute for Forest, Snow and Landscape Research (WSL). https://swissfungi.wsl.ch/de/verbreitungsdaten/verbreitungsatlas/

Gupta, B. S., Jelle, B. P., & Gao, T. (2015). Application of ATR-FTIR spectroscopy to compare the Cell Materials of wood decay fungi with wood mould fungi. *International Journal of Spectroscopy*, *2015*(1), 521938. https://doi.org/10.1155/2015/521938

Gutiérrez, A., del Río, J. C., Martínez, M. J., & Martínez, A. T. (1999). Fungal degradation of lipophilic extractives in *Eucalyptus globulus* wood. *Applied and Environmental Microbiology*, *65*(4), 1367–1371. https://doi.org/10.1128/aem.65.4.1367-1371.1999

Gutiérrez, A., del Río, J. C., Martínez-Íñigo, M. J., Martínez, M. J., & Martínez, Á. T. (2002). Production of new unsaturated lipids during wood decay by ligninolytic basidiomycetes. *Applied and Environmental Microbiology*, *68*(3), 1344–1350. https://doi.org/10.1128/AEM.68.3.1344-1350.2002

Hadrović, S., Braunović, S., Ćirković-Mitrović, T., Eremija, S., Lučić, A., Rakonjac, L., & Jovanović, F. (2024). Biomass carbon and nitrogen content of hardwoods in Novi Pazar (Serbia). *HortScience*, *59*(8), 1067–1068. https://doi.org/10.21273/HORTSCI17824-24

Hobbie, E. A., Grandy, A. S., & Harmon, M. E. (2020). Isotopic and compositional evidence for carbon and nitrogen dynamics during wood decomposition by saprotrophic fungi. *Fungal Ecology*, *45*, 100915. https://doi.org/10.1016/j.funeco.2020.100915

Hobbie, E. A., Macko, S. A., & Shugart, H. H. (1999). Insights into nitrogen and carbon dynamics of ectomycorrhizal and saprotrophic fungi from isotopic evidence. *Oecologia*, *118*(3), 353–360. https://doi.org/10.1007/s004420050736

Högberg, M. N. (2006). Discrepancies between ergosterol and the phospholipid fatty acid 18:2ω6,9 as biomarkers for fungi in boreal forest soils. *Soil Biology and Biochemistry*, *38*(12), 3431–3435. https://doi.org/10.1016/j.soilbio.2006.06.002

Huang, A., Zhou, Q., Liu, J., Fei, B., & Sun, S. (2008). Distinction of three wood species by Fourier transform infrared spectroscopy and two-dimensional correlation IR spectroscopy. *Journal of Molecular Structure*, *883–884*, 160–166. https://doi.org/10.1016/j.molstruc.2007.11.061

Jolly, E. (2018). Pymer4: Connecting R and Python for Linear Mixed Modelling. *Journal of Open Source Software*, *3*(31), 862. https://doi.org/10.21105/joss.00862

Jones, J. M., Heath, K. D., Ferrer, A., & Dalling, J. W. (2020). Habitat-specific effects of bark on wood decomposition: Influences of fragmentation, nitrogen concentration and microbial community composition. *Functional Ecology*, *34*(5), 1123–1133. https://doi.org/10.1111/1365-2435.13547

Jurgensen, M. F., Larsen, M. J., Wolosiewicz, M., & Harvey, A. E. (1989). A comparison of dinitrogen fixation rates in wood litter decayed by white-rot and brown-rot fungi. *Plant and Soil*, *115*(1), 117–122. https://doi.org/10.1007/BF02220701

Karppanen, O., Venäläinen, M., Harju, A. M., & Laakso, T. (2008). The effect of brown-rot decay on water adsorption and chemical composition of Scots pine heartwood. *Annals of Forest Science*, *65*(6), 610–610. https://doi.org/10.1051/forest:2008035

Kiseleva, I. S., Ermoshin, A. A., Galishev, B. A., Shen, D., Ma, C., & He, J. (2022). Chemical composition and antioxidant activity of *Fomitopsis pinicola* growing on coniferous and deciduous substrates. *Forestry Ideas*, *28*(2), 287–297.

Klug, P., & Lewald-Brudi, M. (2023). *Holzzersetzende Pilze* (4th ed.). Arbus Verlag.

Lamlom, S. H., & Savidge, R. A. (2003). A reassessment of carbon content in wood: Variation within and between 41 North American species. *Biomass and Bioenergy*, *25*(4), 381–388. https://doi.org/10.1016/S0961-9534(03)00033-3

Lendzian, K. J., & Beck, A. (2021). Barrier properties of fungal fruit body skins, pileipelles, contribute to protection against water loss. *Scientific Reports*, *11*(1), 8736. https://doi.org/10.1038/s41598-021-88148-0

Li, T., Cui, L., Song, X., Cui, X., Yulian, W., Tang, L., Mu, Y., & Xu, Z. (2022). Wood decay fungi: An analysis of worldwide research. *Journal of Soils and Sediments*, *22,* 1688-1702. https://doi.org/10.1007/s11368-022-03225-9

Lodi, R. S., Dong, X., Wang, X., Han, Y., Liang, X., Peng, C., & Peng, L. (2025). Current research on the medical importance of *Trametes* species. *Fungal Biology Reviews*, *51*, 100413. https://doi.org/10.1016/j.fbr.2025.100413

Lubbers, R. J. M. (2025). An updated perspective on the aromatic metabolic pathways of plant-derived homocyclic aromatic compounds in *Aspergillus niger*. *Microorganisms*, *13*(8), 1718. https://doi.org/10.3390/microorganisms13081718

Mali, T. (2021). *Interactions of Basidiomycota brown rot and white rot fungi: Temporal and spatial effects in decay metabolism and wood degradation* (Doctoral dissertation, University of Helsinki). Helda Repository.

Martínez, A. T., Rencoret, J., Nieto, L., Jiménez-Barbero, J., Gutiérrez, A., & del Río, J. C. (2011). Selective lignin and polysaccharide removal in natural fungal decay of wood as evidenced by in situ structural analyses. *Environmental Microbiology*, *13*(1), 96–107. https://doi.org/10.1111/j.1462-2920.2010.02312.

Md Salim, R., Asik, J., & Sarjadi, M. S. (2021). Chemical functional groups of extractives, cellulose and lignin extracted from native *Leucaena leucocephala* bark. *Wood Science and Technology*, *55*(2), 295–313. https://doi.org/10.1007/s00226-020-01258-2

Micó, E., Aguirrebengoa, M., Quinto, J., Juárez, M., Marmaneu, J., & Sánchez, A. (2024). Physical decomposition stage and ergosterol content predict the chemical composition of downed dead wood in Mediterranean dehesas. *European Journal of Forest Research*, *143*(4), 1117–1133. https://doi.org/10.1007/s10342-024-01672-2

Mille-Lindblom, C., von Wachenfeldt, E., & Tranvik, L. J. (2004). Ergosterol as a measure of living fungal biomass: Persistence in environmental samples after fungal death. *Journal of Microbiological Methods*, *59*(2), 253–262. https://doi.org/10.1016/j.mimet.2004.07.010

Moron, J. B., Klink, G. P. M. van, & Gruter, G.-J. M. (2025). Lignocellulose saccharification: Historical insights and recent industrial advancements towards 2nd generation sugars. *RSC Sustainability*, *3*(3), 1170–1211. https://doi.org/10.1039/D4SU00600C

Nomberg, G., Marinov, O., Arya, G. C., Manasherova, E., & Cohen, H. (2022). The key enzymes in the suberin biosynthetic pathway in plants: An update. *Plants*, *11*(3), 392. https://doi.org/10.3390/plants11030392

Oberg, A. L., & Mahoney, D. W. (2007). Linear Mixed Effects Models. In W. T. Ambrosius (Ed.), *Topics in Biostatistics* (pp. 213–234). Humana Press. https://doi.org/10.1007/978-1-59745-530-5_11

Ofiti, N. O. E., Zosso, C. U., Soong, J. L., Solly, E. F., Torn, M. S., Wiesenberg, G. L. B., & Schmidt, M. W. I. (2021). Warming promotes loss of subsoil carbon through accelerated degradation of plant-derived organic matter. *Soil Biology and Biochemistry*, *156*, 108185. https://doi.org/10.1016/j.soilbio.2021.108185

Özgenc, Ö., Durmaz, S., & Süleyman, K. (2017). Chemical Analysis of Tree Barks using ATR-FTIR Spectroscopy and Conventional Techniques. *Bioresources, 12*(4), 9143–9151. https://doi.org/10.15376/biores.12.4.9143-9151

Pandey, K., & Pitman, A. (2003). FTIR studies of the changes in wood chemistry following decay by brown-rot and white-rot fungi. *International Biodeterioration & Biodegradation*, *52*, 151–160. https://doi.org/10.1016/S0964-8305(03)00052-0

Reich, M., Göbel, C., Kohler, A., Buée, M., Martin, F., Feussner, I., & Polle, A. (2009). Fatty acid metabolism in the ectomycorrhizal fungus *Laccaria bicolor*. *New Phytologist*, *182*(4), 950–964. https://doi.org/10.1111/j.1469-8137.2009.02819.x

Rese, M., van Erven, G., Veersma, R. J., Alfredsen, G., Eijsink, V. G. H., Kabel, M. A., & Tuveng, T. R. (2025). Detailed characterisation of the conversion of hardwood and softwood lignin by a Brown-rot basidiomycete. *Biomacromolecules*, *26*(2), 1063–1074. https://doi.org/10.1021/acs.biomac.4c01403

Ristinmaa, A. S., Korotkova, E., Arntzen, M. Ø., G. H. Eijsink, V., Xu, C., Sundberg, A., Hasani, M., & Larsbrink, J. (2024). Analyses of long-term fungal degradation of spruce bark reveal varying potential for catabolism of polysaccharides and extractive compounds. *Bioresource Technology*, *402*, 130768. https://doi.org/10.1016/j.biortech.2024.130768

Rösecke, J., & König, W. A. (2000). Constituents of various wood-rotting basidiomycetes. *Phytochemistry*, *54*(6), 603–610. https://doi.org/10.1016/S0031-9422(00)00165-5

Rowell, R. M. (Ed.). (2012). *Handbook of Wood Chemistry and Wood Composites* (2nd ed.). CRC Press. https://doi.org/10.1201/b12487

Saranpää, P., & Nyberg, H. (1987). Lipids and sterols of *Pinus sylvestris L.* sapwood and heartwood. *Trees*, *1*(2), 82–87. https://doi.org/10.1007/BF00203575

Schober, P., Boer, C., & Schwarte, L. A. (2018). Correlation coefficients: Appropriate use and interpretation. *Anesthesia & Analgesia*, *126*(5), 1763. https://doi.org/10.1213/ANE.0000000000002864

Schultz, T. P., Goodell, B., & Nicholas, D. D. (2014). Current Understanding of Brown-rot fungal biodegradation mechanisms: A Review. In *ACS Symposium Series (*Vol. 1158, pp. 1–21). American Chemical Society. https://doi.org/10.1021/bk-2014-1158.ch001

Schwarze, F. W. M. R. (2007). Wood decay under the microscope. *Fungal Biology Reviews*, *21*(4), 133–170. https://doi.org/10.1016/j.fbr.2007.09.001

Singara Charya, M. A. (2015). Fungi: An overview. In B. Bahadur, M. Venkat Rajam, L. Sahijram, & K. V. Krishnamurthy (Eds), *Plant Biology and Biotechnology: Volume I: Plant Diversity, Organization, Function and Improvement* (pp. 197–215). Springer India. https://doi.org/10.1007/978-81-322-2286-6_7

Srivastava, Dr. S., Kumar, R., & Singh, V. (2013). Wood decaying fungi. In C. Sevenco (Ed.), *Wood Decaying Fungi*. Lambert Academic Publishing.

Sudharsan, M. S., Rajagopal, K., & Banu, N. (2023). An insight into fungi in forest ecosystems. In Y. M. Rashad, Z. A. M. Baka, & T. A. A. Moussa (Eds), *Plant Mycobiome: Diversity, Interactions and Uses* (pp. 291–318). Springer International Publishing. https://doi.org/10.1007/978-3-031-28307-9_12

Swift, M. J. (1977). The ecology of wood decomposition. *Science Progress (1933-)*, *64*(254), 175–199.

Synytsya, A., Bleha, R., Skrynnikova, A., Babayeva, T., Čopíková, J., Kvasnička, F., Jablonsky, I., & Klouček, P. (2023). Mid-Infrared spectroscopic study of cultivating medicinal fungi *Ganoderma: Composition*, development, and strain variability of Basidiocarps. *Journal of Fungi*, *10*(1), 23. https://doi.org/10.3390/jof10010023

Thimonier, A., Kosonen, Z., Braun, S., Rihm, B., Schleppi, P., Schmitt, M., Seitler, E., Waldner, P., & Thöni, L. (2019). Total deposition of nitrogen in Swiss forests: Comparison of assessment methods and evaluation of changes over two decades. *Atmospheric Environment*, *198*, 335–350. https://doi.org/10.1016/j.atmosenv.2018.10.051

Tláskal, V., Brabcová, V., Větrovský, T., Jomura, M., López-Mondéjar, R., Oliveira Monteiro, L. M., Saraiva, J. P., Human, Z. R., Cajthaml, T., Nunes da Rocha, U., & Baldrian, P. (2021). Complementary roles of wood-inhabiting fungi and bacteria facilitate deadwood decomposition. *mSystems*, *6*(1), e01078-20. https://doi.org/10.1128/msystems.01078-20

Trocha, L. K., Rudy, E., Chen, W., Dabert, M., & Eissenstat, D. M. (2016). Linking the respiration of fungal sporocarps with their nitrogen concentration: Variation among species, tissues and guilds. *Functional Ecology*, *30*(11), 1756–1768. https://doi.org/10.1111/1365-2435.12688

University of Cambridge. (2004). *The structure and mechanical behaviour of wood*. https://www.doitpoms.ac.uk/tlplib/wood/structure_wood_pt2.php

Vane, C. H., Drage, T. C., & Snape, C. E. (2006). Bark decay by the white-rot fungus *Lentinula edodes*: Polysaccharide loss, lignin resistance and the unmasking of suberin. *International Biodeterioration & Biodegradation*, *57*(1), 14–23. https://doi.org/10.1016/j.ibiod.2005.10.004

Waggoner, D. C., Chen, H., Willoughby, A. S., & Hatcher, P. G. (2015). Formation of black carbon-like and alicyclic aliphatic compounds by hydroxyl radical-initiated degradation of lignin. *Organic Geochemistry*, *82*, 69–76. https://doi.org/10.1016/j.orggeochem.2015.02.007

Watkinson, S., Bebber, D., Darrah, P., Fricker, M., Tlalka, M., & Boddy, L. (2006). The role of wood decay fungi in the carbon and nitrogen dynamics of the forest floor. In G. M. Gadd (Ed.), *Fungi in Biogeochemical Cycles* (1st ed., pp. 151–181). Cambridge University Press. https://doi.org/10.1017/CBO9780511550522.008

Webster, R. (2001). Statistics to support soil research and their presentation. *European Journal of Soil Science*, *52*(2), 331–340. https://doi.org/10.1046/j.1365-2389.2001.00383.x

Wehrli, L. (2006). *Lägern Süd: Forschungsstandort und Naturraum. Exkursionsführer.*(Diplomarbeit, University of Zurich).

Wiesenberg, G. L. B., & Gocke, M. I. (2015). Analysis of lipids and polycyclic aromatic hydrocarbons as indicators of past and present (Micro)Biological activity. In T. J. McGenity, K. N. Timmis, & B. Nogales (Eds), *Hydrocarbon and Lipid Microbiology Protocols: Petroleum, Hydrocarbon and Lipid Analysis* (pp. 61–91). Springer. https://doi.org/10.1007/8623_2015_157

Xie, Z.-F., Zhang, W.-W., Zhao, S.-Y., Zhang, X.-H., You, S.-N., Liu, C.-M., & Zhang, G.-Q. (2025). Isolation and structural characterization of melanins from red and yellow varieties of *stropharia rugosoannulata*. *International Journal of Molecular Sciences*, *26*(14), 6985. https://doi.org/10.3390/ijms26146985

Xu, G., Wang, L., Liu, J., & Wu, J. (2013). FTIR and XPS analysis of the changes in bamboo chemical structure decayed by white-rot and brown-rot fungi. *Applied Surface Science*, *280*, 799–805. https://doi.org/10.1016/j.apsusc.2013.05.065

Xue, J., Wei, Y., Chen, L., & Yuan, H. (2025). Transcriptomic insights into the degradation mechanisms of *Fomitopsis pinicola* and its host preference for coniferous over broadleaf deadwood. *Microorganisms*, *13*(5), 1006. https://doi.org/10.3390/microorganisms13051006

9 Appendix

9.1 Tables

Table 2: Sampling strategy protocol.

Fungal specimen	Status	Location	Rot type	Collected on substrate type	Replicates
F. pinicola	*certain*	*Site 1, 2*	*Brown-rot*	*Hardwood, Softwood*	*3, 4 (2 Pseudoreplicates)*
Fomes fomentarius	*certain*	*Site 1*	*White-rot*	*Hardwood*	*3 (1 Pseudoreplicate)*
Trametes versicolor	*certain*	*Site 1,2*	*White-rot*	*Hardwood*	*3*
Trametes sp.	*uncertain*	*Site 1, 2*	*White-rot*	*Hardwood*	*3*

9.1.1 Correlation analysis

Table 3: Results for the correlation analysis.

Correlations	spearman rho	p-value	fisher's z-score	p-value fisher	n
Ergosterol vs. $C_{18:2}$	0.88	< 0.001	-0.515	0.606	38
Ergosterol vs. $C_{18:1}$	0.86	< 0.001	0.120	0.904	44
Ergosterol vs. $C_{26:1}$	0.82	< 0.001	-0.364	0.715	19
Ergosterol vs. $C_{16:1}$	0.78	< 0.001	-0.485	0.627	40

9.1.2 Linear mixed effects models (LMEMs)

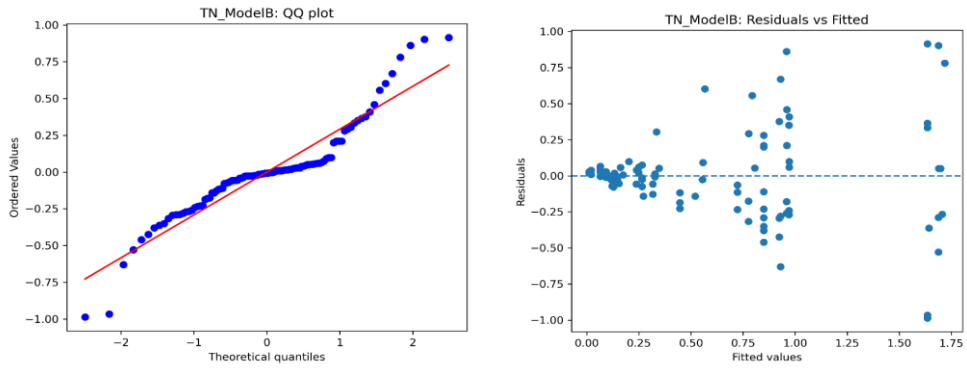

Figure 24: TN values have been log-transformed after identifying non-linearity and heteroscedasticity (right) of the samples in all three models.

Table 4: Overview table of the variables used for the LMEM. The model type indicates which model(s) was/were used for the variable.

Variable	Model type	Model descritpion
TC concentrations	A, B, C	log_TC ~ compartment + (1|Tree_ID) log_TC ~ compartment * rot type + (1|Tree_ID) log_TC ~ compartment * substrate type + (1|Tree_ID)
TN concentrations	A, B, C	log_TN ~ compartment + (1|Tree_ID) log_TN ~ compartment * rot type + (1|Tree_ID) log_TN ~ compartment * substrate type + (1|Tree_ID)
$\delta^{13}C$	A, B, C	$\delta^{13}C$ ~ compartment + (1|Tree_ID) $\delta^{13}C$ ~ compartment * rot type + (1|Tree_ID) $\delta^{13}C$ ~ compartment * substrate type + (1|Tree_ID)
$\delta^{15}N$	A, B, C	$\delta^{15}N$ ~ compartment + (1|Tree_ID) $\delta^{15}N$ ~ compartment * rot type + (1|Tree_ID) $\delta^{15}N$ ~ compartment * substrate type + (1|Tree_ID)
Aromatic C=C (range 1)	A, B, C	AromaticCC(range 1) ~ compartment + (1|Tree_ID) AromaticCC(range 1) ~ compartment * rot type + (1|Tree_ID) AromaticCC(range 1) ~ compartment * substrate type + (1|Tree_ID)
Aromatic C=C (range 2)	A, B, C	AromaticCC(range 2) ~ compartment + (1|Tree_ID) AromaticCC(range 2) ~ compartment * rot type + (1|Tree_ID) AromaticCC(range 2) ~ compartment * substrate type + (1|Tree_ID)
Aliphatic C-H	A, B, C	AliphaticCH ~ compartment + (1|Tree_ID) AliphaticCH ~ compartment * rot type + (1|Tree_ID) AliphaticCH ~ compartment * substrate type + (1|Tree_ID)
Aromatic C-H	A, B, C	log_AromaticCH ~ compartment + (1|Tree_ID) log_AromaticCH ~ compartment * rot type + (1|Tree_ID) log_AromaticCH ~ compartment * substrate type + (1|Tree_ID)
Carboxylic C=O	A, B, C	CarboxylicCO ~ compartment + (1|Tree_ID) CarboxylicCO ~ compartment * rot type + (1|Tree_ID) CarboxylicCO ~ compartment * substrate type + (1|Tree_ID)
Lignin-like residues	A, B, C	Lignin ~ compartment + (1|Tree_ID) Lignin ~ compartment * rot type + (1|Tree_ID) Lignin ~ compartment * substrate type + (1|Tree_ID)
Polysaccharide	A, B, C	Polysaccharide ~ compartment + (1|Tree_ID) Polysaccharide ~ compartment * rot type + (1|Tree_ID) Polysaccharide ~ compartment * substrate type + (1|Tree_ID)
Cellulose	A, B, C	Cellulose ~ compartment + (1|Tree_ID) Cellulose ~ compartment * rot type + (1|Tree_ID) log_Cellulose ~ compartment * substrate type + (1|Tree_ID)
Saturated/unsaturated FA ratio	A	log_SFA/UFA ratio ~ compartment + (1|Tree_ID)
Long-chain/short-chain FA ratio	A	log_LCFA/SCFA ratio ~ compartment + (1|Tree_ID)
BCFA	A	log_BCFA ~ compartment + (1|Tree_ID)

DCA	A	log_DCA ~ compartment + (1|Tree_ID)
SSCFA	A	log_SSCFA ~ compartment + (1|Tree_ID)
SLCFA	A	log_SLCFA ~ compartment + (1|Tree_ID)
USCFA	A	log_USCFA ~ compartment + (1|Tree_ID)
ULCFA	A	log_ULCFA ~ compartment + (1|Tree_ID)
$C_{18:1}$	C	log_$C_{18:1}$ ~ compartment + (1|Tree_ID)
$C_{18:2}$	C	log_$C_{18:2}$ ~ compartment + (1|Tree_ID)
$C_{16:1}$	C	log_$C_{16:1}$ ~ compartment + (1|Tree_ID)
$C_{26:1}$	C	log_$C_{26:1}$ ~ compartment + (1|Tree_ID)
ergosterol	A, C	log_ergosterol ~ compartment + (1|Tree_ID) log_ergosterol ~ compartment * substrate type + (1|Tree_ID)
erg. 2 (dehydroergosterol)	A	log_erg.2 (dehydroergosterol) ~ compartment + (1|Tree_ID)
stigmastanol	A	log_stigmastanol ~ compartment + (1|Tree_ID)
β-sitosterol	A	log_ β-sitosterol ~ compartment + (1|Tree_ID)
Ɣ-sitostenone	A	log_ Ɣ-sitostenone ~ compartment + (1|Tree_ID)

*Table 5: **Significant** ANOVA results for the fixed effects of the LMEMs.*

Variable	Effect	NumDF	DenomDF	F value	p-value
TC concentrations	Compartment	4	92.56	8.63	< 0.001
	Rot type	1	17.39	15.15	< 0.01
	Compartment: Rot type interaction	4	83.14	3.79	< 0.01
TN concentrations	Compartment	4	94.10	223.45	< 0.001
$\delta^{13}C$	Compartment	4	93.65	55.58	< 0.001
$\delta^{15}N$	Compartment	4	95.13	62.70	< 0.001
	Compartment:Rot type interaction	4	89.92	3.32	< 0.05
Aromatic C=C (range 1)	Compartment	4	104.0	15.48	< 0.001
	Rot type	1	38.36	4.38	< 0.05
	Compartment:Rot type interaction	4	89.90	20.50	< 0.001
	Compartment:Substrate type interaction	6	36.57	3.91	< 0.01
Aromatic C=C (range 2)	Compartment	4	95.56	71.54	< 0.001
	Compartment:Rot type interaction	4	90.01	3.07	< 0.05
	Compartment:Substrate type interaction	6	36.07	3.60	< 0.01
Aliphatic C-H	Compartment	4	100.34	83.53	< 0.001
	Compartment:Rot type interaction	4	99.0	27.83	< 0.001
Aromatic C-H	Compartment	4	95.55	47.06	< 0.001

	Rot type	1	40.68	9.70	< 0.01
	Compartment:Rot type interaction	4	88.88	5.47	< 0.001
	Substrate type	1	1.97	24.88	< 0.05
	Compartment:Substrate type interaction	6	35.83	11.22	< 0.001
Carboxylic C=O	Compartment	4	104.0	62.68	< 0.001
	Compartment:Rot type interaction	4	91.48	21.86	< 0.001
	Substrate type	1	3.70	10.60	< 0.05
	Compartment:Substrate type interaction	6	37.22	22.26	< 0.001
Lignin-like residues	Compartment	4	95.81	274.09	< 0.001
	Rot type	1	63.06	8.52	< 0.01
	Compartment:Rot type interaction	4	88.62	7.31	< 0.001
	Substrate type	1	40.0	44.43	< 0.001
	Compartment:Substrate type interaction	6	40.0	5.47	< 0.001
Polysaccharide	Compartment	4	96.32	20.25	< 0.001
	Rot type	1	38.00	9.18	< 0.01
Cellulose	Compartment	4	95.98	58.70	< 0.001
	Compartment:Rot type interaction	4	88.57	8.82	< 0.001
	Substrate type	1	3.21	76.14	< 0.01
	Compartment:Substrate type interaction	6	35.93	35.14	< 0.001
Saturated/unsaturated FA ratio	Compartment	4	42.74	42.75	< 0.001
Long-chain/short-chain FA ratio	Compartment	4	43.00	82.25	< 0.001
BCFA	Compartment	4	42.24	6.79	< 0.001
DCA	Compartment	4	43.71	4.78	< 0.01
SSCFA	Compartment	4	43.03	11.62	< 0.001
SLCFA	Compartment	4	44.04	24.08	< 0.001
USCFA	Compartment	4	43.01	44.33	< 0.001
ULCFA	Compartment	4	43.33	30.26	< 0.001
$C_{18:1}$	Compartment	6	36.29	29.86	< 0.001
$C_{16:1}$	Compartment	6	39.00	7.32	< 0.001
$C_{26:1}$	Compartment	6	34.81	33.66	< 0.001
ergosterol	Compartment	4	43.62	15.67	< 0.001
	Compartment:Substrate type interaction	6	34.73	4.20	< 0.01

erg. 2 (dehydroergosterol)	Compartment	4	41.80	15.70	< 0.001
stigmastanol	Compartment	4	43.34	2397.37	< 0.001
β-sitosterol	Compartment	4	43.50	3603.60	< 0.001
ɣ-sitostenone	Compartment	4	43.53	1619.37	< 0.001

*Table 6: **Significant** post-hoc results from **model type A**.*

Variable	Model type	Compartment group	Estimate	SE	t-ratio	p-value
TC concentrations	A	Bark – wood with mycelia	-0.059	0.019	-3.06	< 0.05
	A	Basidiocarp – weathered wood	-0.064	0.015	-4.247	< 0.001
	A	Basidiocarp – wood with mycelia	-0.079	0.016	-5.034	< 0.001
	A	Basidiocarp – wood without mycelia	-0.046	0.015	-3.004	< 0.05
TN concentrations	A	Bark - basidiocarp	-0.695	0.114	-6.119	< 0.001
	A	Bark – weathered wood	0.955	0.129	7.398	< 0.001
	A	Bark – wood with mycelia	1.807	0.132	13.711	< 0.001
	A	Bark – wood without mycelia	1.900	0.131	14.501	< 0.001
	A	Basidiocarp – weathered wood	1.650	0.104	15.806	< 0.001
	A	Basidiocarp – wood with mycelia	2.501	0.108	23.163	< 0.001
	A	Basidiocarp – wood without mycelia	2.595	0.107	24.277	< 0.001
	A	Weathered wood – wood with mycelia	0.852	0.123	6.949	< 0.001
	A	Weathered wood – wood without mycelia	0.945	0.121	7.823	< 0.001
$\delta^{13}C$	A	Bark - basidiocarp	-2.924	0.294	-9.963	< 0.001
	A	Basidiocarp – weathered wood	2.694	0.270	9.986	< 0.001
	A	Basidiocarp – wood with mycelia	3.021	0.279	10.823	< 0.001
	A	Basidiocarp – wood without mycelia	2.987	0.276	10.809	< 0.001
$\delta^{15}N$	A	Bark - basidiocarp	1.218	0.453	2.691	< 0.05
	A	Bark – wood with mycelia	-3.880	0.526	-7.373	< 0.001
	A	Bark – wood without mycelia	-4.467	0.523	-8.544	< 0.001
	A	Basidiocarp – weathered wood	-2.326	0.416	-5.587	< 0.001
	A	Basidiocarp – wood with mycelia	-5.099	0.431	-11.830	< 0.001
	A	Basidiocarp – wood without mycelia	-5.686	0.426	-13.346	< 0.001
	A	Weathered wood – wood without mycelia	-2.772	0.491	-5.644	< 0.001

	A	Weathered wood – wood without mycelia	-3.360	0.484	-6.937	< 0.001
Aliphatic C-H	A	Bark - basidiocarp	-24.042	3.582	-6.712	< 0.001
	A	Bark – weathered wood	-53.258	4.085	-13.037	< 0.001
	A	Bark – wood with mycelia	-56.779	4.185	-13.566	< 0.001
	A	Bark – wood without mycelia	-58.089	4.137	-14.042	< 0.001
	A	Basidiocarp – weathered wood	-29.216	3.296	-8.864	< 0.001
	A	Basidiocarp - wood with mycelia	-32.736	3.421	-9.569	< 0.001
	A	Basidiocarp - wood without mycelia	-34.046	3.360	-10.132	< 0.001
Aromatic C=C (range 2)	A	Bark – weathered wood	10.424	1.411	-7.151	< 0.001
	A	Bark – wood with mycelia	13.342	1.442	-7.259	< 0.001
	A	Bark – wood without mycelia	13.551	1.431	-7.906	< 0.001
	A	Basidiocarp – weathered wood	11.534	1.139	-10.877	< 0.001
	A	Basidiocarp - wood with mycelia	14.453	1.180	-10.818	< 0.001
	A	Basidiocarp - wood without mycelia	14.662	1.164	-11.714	< 0.001
Aromatic C=C (range 1)	A	Bark - basidiocarp	-1.937	0.274	-7.081	< 0.001
	A	Bark – weathered wood	-2.027	0.312	-6.494	< 0.001
	A	Bark – wood with mycelia	-2.024	0.320	-6.324	< 0.001
	A	Bark – wood without mycelia	-1.953	0.316	-6.183	< 0.001
Aromatic C-H	A	Bark - basidiocarp	3.112	0.248	12.533	< 0.001
	A	Bark – weathered wood	2.988	0.283	10.572	< 0.001
	A	Bark – wood with mycelia	3.300	0.289	11.428	< 0.001
	A	Bark – wood without mycelia	3.087	0.287	10.763	< 0.001
Carboxylic C=O	A	Bark – weathered wood	-12.522	1.751	-7.151	< 0.001
	A	Bark – wood with mycelia	-13.037	1.796	-7.259	< 0.001
	A	Bark – wood without mycelia	-14.013	1.772	-7.906	< 0.001
	A	Basidiocarp – weathered wood	-15.364	1.413	-10.877	< 0.001
	A	Basidiocarp - wood with mycelia	-15.880	1.468	-10.818	< 0.001
	A	Basidiocarp - wood without mycelia	-16.855	1.439	-11.714	< 0.001
Lignin-like residues	A	Bark - basidiocarp	6.379	0.713	8.953	< 0.001
	A	Bark – weathered wood	-8.049	0.811	-9.929	< 0.001
	A	Bark – wood with mycelia	-10.922	0.828	-13.195	< 0.001
	A	Bark – wood without mycelia	-10.638	0.823	-12.931	< 0.001
	A	Basidiocarp – weathered wood	-14.428	0.655	-22.023	< 0.001
	A	Basidiocarp - wood with mycelia	-17.302	0.678	-25.517	< 0.001
	A	Basidiocarp - wood without mycelia	-17.018	0.671	-25.377	< 0.001

	A	Weathered wood – wood with mycelia	-2.874	0.772	-3.724	< 0.01
	A	Weathered wood – wood without mycelia	-2.590	0.761	-3.404	< 0.01
Polysaccharide	A	Bark - basidiocarp	0.760	0.239	3.184	< 0.01
	A	Bark – weathered wood	1.634	0.272	6.008	< 0.001
	A	Bark – wood with mycelia	1.952	0.278	7.015	< 0.001
	A	Bark – wood without mycelia	1.876	0.276	6.806	< 0.001
	A	Basidiocarp – weathered wood	0.874	0.219	3.983	< 0.001
	A	Basidiocarp - wood with mycelia	1.192	0.227	5.239	< 0.001
	A	Basidiocarp - wood without mycelia	1.116	0.224	4.980	< 0.001
Cellulose	A	Bark – weathered wood	-10.495	1.237	-8.487	< 0.001
	A	Bark – wood with mycelia	-10.444	1.263	-8.271	< 0.001
	A	Bark – wood without mycelia	-9.623	1.255	-7.667	< 0.001
	A	Basidiocarp – weathered wood	-11.137	1.000	-11.142	< 0.001
	A	Basidiocarp - wood with mycelia	-11.086	1.034	-10.717	< 0.001
	A	Basidiocarp - wood without mycelia	-10.265	1.023	-10.032	< 0.001
Saturated/unsaturated FA ratio	A	Bark - basidiocarp	3.349	0.343	9.755	< 0.001
	A	Basidiocarp – weathered wood	-2.401	0.325	-7.395	< 0.001
	A	Basidiocarp – wood with mycelia	-2.936	0.325	-9.044	< 0.001
	A	Basidiocarp – wood without mycelia	-2.660	0.343	-7.748	< 0.001
Long-chain/short-chain FA ratio	A	Bark - basidiocarp	4.011	0.245	16.370	< 0.001
	A	Bark – weathered wood	1.840	0.293	6.275	< 0.001
	A	Bark – wood with mycelia	1.755	0.293	5.987	< 0.001
	A	Bark – wood without mycelia	2.015	0.304	6.635	< 0.001
	A	Basidiocarp – weathered wood	-2.171	0.232	-9.361	< 0.001
	A	Basidiocarp - wood with mycelia	-2.255	0.232	-9.725	< 0.001
	A	Basidiocarp - wood without mycelia	-1.995	0.245	-8.144	< 0.001
BCFA	A	Bark - basidiocarp	4.764	1.588	3.000	< 0.05
	A	Bark – weathered wood	8.280	1.900	4.358	< 0.001
	A	Bark – wood with mycelia	7.441	1.900	3.917	< 0.01
	A	Bark – wood without mycelia	8.199	1.968	4.165	< 0.01
DCA	A	Bark - basidiocarp	7.536	1.852	4.069	< 0.01
SSCFA	A	Bark - basidiocarp	-0.486	0.166	-2.921	< 0.05
	A	Basidiocarp – weathered wood	0.701	0.166	4.460	< 0.001
	A	Basidiocarp - wood with mycelia	0.816	0.157	5.194	< 0.001
	A	Basidiocarp - wood without mycelia	0.778	0.157	4.679	< 0.001

SLCFA	A	Bark - basidiocarp	2.275	0.240	9.471	< 0.001
	A	Bark – weathered wood	1.927	0.287	6.704	< 0.001
	A	Bark – wood with mycelia	2.146	0.287	7.466	< 0.001
	A	Bark – wood without mycelia	2.252	0.298	7.560	< 0.001
USCFA	A	Bark - basidiocarp	-2.942	0.379	-7.761	< 0.001
	A	Basidiocarp – weathered wood	2.931	0.360	8.136	< 0.001
	A	Basidiocarp - wood with mycelia	3.608	0.360	10.016	< 0.001
	A	Basidiocarp - wood without mycelia	3.342	0.379	8.815	< 0.001
ULCFA	A	Bark - basidiocarp	-7.362	1.192	-6.177	< 0.001
	A	Basidiocarp – weathered wood	6.929	1.131	6.128	< 0.001
	A	Basidiocarp - wood with mycelia	8.929	1.131	7.898	< 0.001
	A	Basidiocarp - wood without mycelia	9.770	1.192	8.197	< 0.001
ergosterol	A	Bark - basidiocarp	-10.851	1.541	-7.043	< 0.001
	A	Bark – wood with mycelia	-6.868	1.817	-3.780	< 0.01
	A	Bark – wood without mycelia	-8.254	1.880	-4.390	< 0.001
	A	Basidiocarp – weathered wood	6.813	1.363	4.998	< 0.001
	A	Basidiocarp - wood with mycelia	3.982	1.363	2.921	< 0.05
erg. 2 (dehydroergosterol)	A	Bark - basidiocarp	-10.133	1.375	-7.368	< 0.001
	A	Bark – weathered wood	-5.015	1.625	-3.086	< 0.05
	A	Bark – wood with mycelia	-6.099	1.625	-3.753	< 0.01
	A	Bark – wood without mycelia	-7.986	1.677	-4.763	< 0.001
	A	Basidiocarp – weathered wood	5.118	1.224	4.180	< 0.01
	A	Basidiocarp - wood with mycelia	4.034	1.224	3.295	< 0.05
stigmastanol	A	Bark – weathered wood	1.258	0.190	6.631	< 0.001
	A	Bark – wood with mycelia	1.339	0.190	7.059	< 0.001
	A	Bark – wood without mycelia	1.599	0.196	8.175	< 0.001
β-sitosterol	A	Bark – weathered wood	1.545	0.261	5.913	< 0.001
	A	Bark – wood with mycelia	1.976	0.261	7.563	< 0.001
	A	Bark – wood without mycelia	2.161	0.270	7.995	< 0.001
γ-sitostenone	A	Bark – weathered wood	1.338	0.294	4.549	< 0.001
	A	Bark – wood with mycelia	0.854	0.294	2.904	< 0.05
	A	Bark – wood without mycelia	1.416	0.305	4.651	< 0.001

*Table 7: **Significant** post-hoc results from **model type B** with WR = White rot and BR = Brown rot.*

Variable	Model type	Compartment group	Estimate	SE	t-ratio	p-value
TC concentrations	B	WR bark – BR bark	-0.064	0.027	-2.370	< 0.05
	B	WR basidiocarp – WR basidiocarp	-0.108	0.018	-5.945	< 0.001
TN concentrations	B	WR basidiocarp – BR basidiocarp	0.328	0.158	2.070	< 0.05
$\delta^{15}N$	B	WR wood with mycelia – BR wood with mycelia	-2.059	0.757	-2.719	< 0.01
Aliphatic C-H	B	WR basidiocarp – BR basidiocarp	-29.738	2.707	-10.987	< 0.001
	B	WR wood with mycelia – BR wood with mycelia	10.143	3.896	2.603	< 0.05
Carboxylic C=O	B	WR basidiocarp – BR basidiocarp	-9.122	1.338	-6.817	< 0.001
	B	WR weathered wood – BR weathered wood	4.378	1.790	2.446	< 0.05
	B	WR wood with mycelia – BR wood with mycelia	8.148	1.867	4.364	< 0.001
	B	WR wood without mycelia – BR wood without mycelia	5.159	1.861	2.772	< 0.01
Aromatic C=C (range 2)	B	WR basidiocarp – BR basidiocarp	4.766	1.464	3.255	< 0.01
Lignin-like residues	B	WR weathered wood – BR weathered wood	-2.881	1.059	-2.721	< 0.01
	B	WR wood with mycelia – BR wood with mycelia	-5.188	1.102	-4.709	< 0.001
	B	WR wood without mycelia – BR wood without mycelia	-2.269	1.093	-2.075	< 0.05
Polysaccharide	B	WR wood with mycelia – BR wood with mycelia	-1.368	0.381	-3.588	< 0.001
Aromatic C=C (range 1)	B	WR basidiocarp – BR basidiocarp	-2.228	0.242	-9.203	< 0.001
Aromatic C-H	B	WR weathered wood – BR weathered wood	-1.309	0.372	-3.516	< 0.001
	B	WR wood with mycelia – BR wood with mycelia	-1.198	0.388	-3.091	< 0.01
	B	WR wood without mycelia – BR wood without mycelia	-1.068	0.386	-2.770	< 0.01
Cellulose	B	WR basidiocarp – BR basidiocarp	-4.811	1.316	-3.655	< 0.001
	B	WR wood with mycelia – BR wood with mycelia	3.548	1.687	2.103	< 0.05

	B	WR wood without mycelia – BR wood without mycelia	3.448	1.668	2.067	< 0.05

*Table 8: **Significant** post-hoc results from **model type C** with HW = Hardwood and SW = Softwood.*

Variable	**Model type**	**Compartment group**	**Estimate**	**SE**	**t-ratio**	**p-value**
TC concentrations	C	HW pileipellis – SW pileipellis	-0.068	0.028	-2.421	< 0.05
	C	HW trama – SW trama	-0.081	0.028	-2.875	< 0.01
TN concentrations	C	HW bark – SW bark	0.827	0.312	2.649	< 0.05
	C	HW weathered wood – SW weathered wood	0.799	0.296	2.695	< 0.05
	C	HW wood without mycelia – SW wood without mycelia	0.674	0.312	2.159	< 0.05
$\delta^{15}N$	C	HW trama – SW trama	3.485	1.355	2.573	< 0.05
Aliphatic C-H	C	HW wood with mycelia – SW wood with mycelia	9.920	4.462	2.223	< 0.05
Carboxylic C=O	C	HW trama – SW trama	-6.094	1.649	-3.696	< 0.001
	C	HW weathered wood – SW weathered wood	10.276	1.649	6.232	< 0.001
	C	HW wood with mycelia – SW wood with mycelia	12.911	1.649	7.831	< 0.001
	C	HW wood without mycelia – SW wood without mycelia	7.127	1.770	4.026	< 0.001
Aromatic C=C (range 2)	C	HW pileipellis – SW pileipellis	8.061	2.695	2.990	< 0.05
	C	HW trama – SW trama	9.048	2.695	3.357	< 0.01
Lignin-like residues	C	HW weathered wood – SW weathered wood	-6.354	1.398	-4.546	< 0.001
	C	HW wood with mycelia – SW wood with mycelia	-8.994	1.398	-6.434	< 0.001
	C	HW wood without mycelia – SW wood without mycelia	-5.327	1.510	-3.528	< 0.01
Aromatic C=C (range 1)	C	HW trama – SW trama	1.159	0.385	3.014	< 0.01
	C	HW weathered wood – SW weathered wood	1.262	0.385	3.281	< 0.01
	C	HW wood with mycelia – SW wood with mycelia	0.984	0.385	2.558	< 0.05
Aromatic C-H	C	HW weathered wood – SW weathered wood	-1.568	0.393	-3.994	< 0.001

	C	HW wood with mycelia – SW wood with mycelia	-2.721	0.393	-6.930	< 0.001
	C	HW wood without mycelia – SW wood without mycelia	-2.346	0.423	-5.544	< 0.001
Cellulose	C	HW bark – SW bark	5.727	1.353	4.233	< 0.001
	C	HW weathered wood – SW weathered wood	16.049	1.275	12.589	< 0.001
	C	HW wood with mycelia – SW wood with mycelia	13.162	1.275	10.324	< 0.001
	C	HW wood without mycelia – SW wood without mycelia	11.416	1.353	8.438	< 0.001
$C_{18:2}$	C	HW hymenium – SW hymenium	7.489	3.335	2.246	< 0.05
$C_{26:1}$	C	HW pileipellis – SW pileipellis	-3.710	1.648	-2.252	< 0.05
ergosterol	C	HW bark – SW bark	-8.512	2.413	-3.527	< 0.01
	C	HW weathered wood – SW weathered wood	-10.211	2.090	-4.887	< 0.001

9.1.3 Principal component analysis (PCA)

*Table 9: Importance of components in the first PCA including the Eigenvalue, % variance and cumulative % of **organic compound group values** including all compartments.*

PC	Standard deviation	Eigenvalue	Proportion of Variance (%)	Cumulative Variance (%)
PC1	2.267	5.143	63.523	63.523
PC2	1.078	1.163	14.370	77.894
PC3	0.898	0.806	9.967	87.861
PC4	0.665	0.442	5.469	93.331
PC5	0.560	0.314	3.878	97.210
PC6	0.365	0.133	1.646	98.856
PC7	0.267	0.071	0.885	99.742
PC8	0.144	0.020	0.257	100.000

*Table 10: First three eigenvectors from the PCA of the standardized **organic compound group values** including all compartments.*

Variable	PC1	PC2	PC3
Aliphatic C-H	-0.429	0.028	-0.053
Carboxylic C=O	-0.406	-0.137	0.338
Aromatic C=C (range 2)	0.373	0.355	0.192
Lignin-like residues	-0.317	-0.559	-0.253
Polysaccharide	0.318	-0.158	0.491

Aromatic C=C (range 1)	-0.335	0.377	0.357
Cellulose	-0.367	0.0719	0.484
Aromatic C-H	0.247	-0.606	0.419

*Table 11: Importance of components in the first PCA including the Eigenvalue, % variance and cumulative % of **organic compound group values** including only **basidiocarp** samples.*

PC	Standard deviation	Eigenvalue	Proportion of Variance (%)	Cumulative Variance (%)
PC1	1.779	3.166	50.143	50.143
PC2	1.268	1.609	25.478	75.621
PC3	0.999	0.999	15.823	91.444
PC4	0.602	0.362	5.744	97.189
PC5	0.407	0.165	2.626	99.816
PC6	0.107	0.011	0.183	100.000

*Table 12: First three eigenvectors from the PCA of the standardized **organic compound group values** including only **basidiocarp** samples.*

Variable	PC1	PC2	PC3
Aromatic C=C (range 1)	0.531	-0.037	0.158
Aromatic C=C (range 2)	-0.304	0.486	0.477
Carboxylic C=O	0.555	0.104	-0.020
Aliphatic C-H	0.558	0.133	0.094
Polysaccharide	0.055	0.734	0.181
Aromatic C-H	-0.011	0.440	-0.839

*Table 13: Importance of components in the first PCA including the Eigenvalue, % variance and cumulative % of **organic compound group values, fatty acid and sterol group values** including all compartments.*

PC	Standard deviation	Eigenvalue	Proportion of Variance (%)	Cumulative Variance (%)
PC1	2.611	6.821	35.214	35.214
PC2	2.400	5.760	29.734	64.948
PC3	1.304	1.702	8.789	73.738
PC4	1.047	1.096	5.662	79.400
PC5	1.005	1.011	5.221	84.622
PC6	0.886	0.785	4.053	88.676
PC7	0.747	0.559	2.885	91.561
PC8	0.690	0.476	2.461	94.023
PC9	0.560	0.314	1.623	95.646
PC10	0.464	0.215	1.112	96.759
PC11	0.446	0.199	1.028	97.787
PC12	0.339	0.115	0.595	98.382
PC13	0.295	0.087	0.450	98.833
PC14	0.282	0.079	0.411	99.245
PC15	0.236	0.055	0.288	99.533
PC16	0.195	0.038	0.197	99.731

PC17	0.166	0.027	0.142	99.874
PC18	0.118	0.014	0.072	99.946
PC19	0.101	0.010	0.053	100.000

*Table 14: First three eigenvectors from the PCA of the standardized **organic compound group values, fatty acid and sterol group values** including all compartments.*

Variable	**PC1**	**PC2**	**PC3**
Aliphatic C-H	-0.202	-0.338	0.084
Carboxylic C=O	0.036	-0.331	0.307
Aromatic C=C (range 2)	0.002	0.352	0.072
Lignin-like residues	0.109	-0.350	-0.189
Polysaccharide	0.121	0.207	0.157
Aromatic C=C (range 1)	-0.297	0.009	0.246
Cellulose	-0.001	-0.279	0.471
Aromatic C-H	0.349	0.124	-0.032
BCFA	0.194	0.181	-0.221
DCA	0.281	0.155	0.299
SSCFA	-0.104	0.270	0.426
SLCFA	0.319	0.158	0.202
USCFA	-0.244	0.277	0.077
ULCFA	-0.214	0.260	0.053
ergosterol	-0.205	0.239	0.009
erg. 2 (dehydroergosterol)	-0.165	0.155	-0.149
stigmastanol	0.353	0.044	0.102
β-sitosterol	0.305	0.056	-0.268
ɣ-sitostenone	0.310	-0.066	0.269

*Table 15: Importance of components in the first PCA including the Eigenvalue, % variance and cumulative % of **organic compound group values, fatty acid and sterol group values** including basidiocarp samples.*

PC	**Standard deviation**	**Eigenvalue**	**Proportion of Variance (%)**	**Cumulative Variance (%)**
PC1	2.03	4.123	32.864	32.864
PC2	1.749	3.062	24.410	57.275
PC3	1.318	1.737	13.849	71.125
PC4	0.972	0.945	7.537	78.662
PC5	0.903	0.817	6.513	85.176
PC6	0.768	0.590	4.703	89.880
PC7	0.741	0.549	4.376	94.257
PC8	0.593	0.352	2.806	97.064
PC9	0.414	0.171	1.370	98.434
PC10	0.307	0.094	0.755	99.189
PC11	0.248	0.061	0.493	99.682
PC12	0.199	0.039	0.317	100.000

*Table 16: First three eigenvectors from the PCA of the standardized **organic compound group values, fatty acid and sterol group values** including basidiocarp samples.*

Variable	PC1	PC2	PC3
Aliphatic C-H	0.054	0.499	-0.163
Carboxylic C=O	0.431	0.218	-0.024
Aromatic C=C (range 2)	-0.235	-0.469	-0.009
Polysaccharide	-0.243	0.109	0.468
Aromatic C=C (range 1)	-0.356	0.134	0.017
Aromatic C-H	0.038	-0.391	0.218
SSCFA	0.076	-0.363	-0.356
SLCFA	0.343	0.080	0.396
USCFA	0.355	-0.219	-0.358
ULCFA	0.383	0.139	0.060
ergosterol	0.405	-0.259	0.213
erg. 2 (dehydroergosterol)	0.076	-0.159	0.492

9.2 Code

9.2.1 Model analyses

```
# ========================================================================
# Linear mixed-effects modelling using pymer4 (Lmer).
# Random structure: random intercept for Tree_ID.
#
# Model A (RQ1): compartment effect
# Model B (RQ2): compartment × rot type
# Model C (RQ3): compartment × substrate type (F. pinicola only)
#
# Fixed effects tested using likelihood ratio tests (ML).
# Final estimates reported using REML.
#
# Transformation choices:
# - TC  -> log-transformed in all models
# - TN  -> log-transformed in all models
# - d13C -> raw
# - d15N -> raw
# ========================================================================

import os
import re
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt
from scipy import stats
from pymer4.models import Lmer

# file paths to fetch the data and store the models
FILE_PATH  = r"C:\Users\Ladina\OneDrive - Universität Zürich UZH\Master\Masterarbeit\EA
data\Calculations_EA_Data_Ladina_Reich.xlsx"
SHEET_NAME = "EA_CN"
OUT_DIR = r"C:\Users\Ladina\OneDrive - Universität Zürich UZH\Master\Masterarbeit\EA
data\model_diagnostics"
os.makedirs(OUT_DIR, exist_ok=True)

COL_TREE   = "Tree_ID"
COL_ROT    = "Rot_type"
COL_SPEC   = "Species_type"
COL_SUB    = "Substrate_type"
COL_COMP   = "Compartment"

# Original names in excel
COL_TC     = "TC concentration [%]"
COL_TN     = "TN concentration [%]"
COL_D13C   = "Delta13C [‰]"
```

```
COL_D15N   = "Delta15N [‰]"
COL_CN     = "CN ratio"

USE_COLS   = [COL_TREE, COL_ROT, COL_SPEC, COL_SUB, COL_COMP, COL_TC, COL_TN, COL_D13C, COL_D15N, COL_CN]
NUM_COLS   = [COL_TC, COL_TN, COL_D13C, COL_D15N, COL_CN]

# safe response names
SAFE_MAP = {
    COL_TC: "TC",
    COL_TN: "TN",
    COL_D13C: "d13C",
    COL_D15N: "d15N",
}
RESPONSES_SAFE = ["TC", "TN", "d13C", "d15N"]

# final transformation choices
FINAL_SCALE = {
    "TC": "log",
    "TN": "log",
    "d13C": "raw",
    "d15N": "raw",
}

ROT_LEVELS = ["white-rot", "brown-rot"]
SUB_LEVELS = ["hardwood", "softwood"]

P_ADJUST = "holm"
REML_FINAL = True
REML_LRT   = False
LOG_OFFSET = 1e-6

# cleaning
def to_num(x):
    if pd.isna(x):
        return np.nan
    s = str(x).strip().replace(",", ".")
    s = "".join(ch for ch in s if ch.isdigit() or ch in ".-+eE")
    return pd.to_numeric(s, errors="coerce")

def normalize_text(s):
    if pd.isna(s):
        return np.nan
    s = str(s).strip()
    s = s.replace("–", "-").replace("—", "-").replace("_", "-")
    s = re.sub(r"\s+", " ", s)
    return s.lower()

def clean(df: pd.DataFrame) -> pd.DataFrame:
    df = df.loc[:, ~df.columns.astype(str).str.startswith("Unnamed")]
    df = df.loc[:, ~df.columns.duplicated(keep="last")]

    keep = [c for c in USE_COLS if c in df.columns]
    df = df[keep].copy()

    for c in NUM_COLS:
        if c in df.columns:
            df[c] = df[c].apply(to_num)

    df[COL_TREE] = df[COL_TREE].astype(str).str.strip()

    for c in [COL_ROT, COL_SUB, COL_COMP, COL_SPEC]:
        if c in df.columns:
            df[c] = df[c].apply(normalize_text)

    if COL_ROT in df.columns:
        df[COL_ROT] = df[COL_ROT].str.replace(" ", "-", regex=False).replace({
            "white rot": "white-rot",
            "brown rot": "brown-rot",
            "white-rot": "white-rot",
            "brown-rot": "brown-rot",
        })

    if COL_SUB in df.columns:
        df[COL_SUB] = df[COL_SUB].replace({"hard wood": "hardwood", "soft wood": "softwood"})

    if COL_SPEC in df.columns:
        df[COL_SPEC] = df[COL_SPEC].replace({
            "f. pinicola": "fomitopsis pinicola",
            "fomitopsis_pinicola": "fomitopsis pinicola",
        })

    return df


# mapping compartments
FRUITING_BODY_ALIASES = {
    "fruiting body",
    "fruiting body outside",
```

```
    "fruiting body inside",
    "fruiting body spores",
    "basidiocarp",
    "sporocarp",
}

WOOD_COMPARTMENTS = {
    "wood with mycelia",
    "wood without mycelia",
    "weathered wood",
    "bark",
}

PINICOLA_FINE_FRUITING = {
    "fruiting body outside": "pileipellis",
    "fruiting body inside": "trama",
    "fruiting body spores": "hymenium",
    "fruiting body": "basidiocarp",
    "basidiocarp": "basidiocarp",
    "sporocarp": "basidiocarp",
}

def map_compartment_collapsed(s):
    if pd.isna(s):
        return np.nan
    s = normalize_text(s)
    if s in FRUITING_BODY_ALIASES or s.startswith("fruiting body"):
        return "basidiocarp"
    if s in WOOD_COMPARTMENTS:
        return s
    return np.nan

def map_compartment_pinicola_fine(s):
    if pd.isna(s):
        return np.nan
    s = normalize_text(s)
    if s in PINICOLA_FINE_FRUITING:
        return PINICOLA_FINE_FRUITING[s]
    if s in WOOD_COMPARTMENTS:
        return s
    if s.startswith("fruiting body outside"):
        return "pileipellis"
    if s.startswith("fruiting body inside"):
        return "trama"
    if s.startswith("fruiting body spores"):
        return "hymenium"
    if s.startswith("fruiting body"):
        return "basidiocarp"
    return np.nan

# helpers
def print_header(title):
    print("\n" + "#" * 110)
    print(title)
    print("#" * 110)

def cell_counts(df, a, b=None):
    if b is None:
        print(df[a].value_counts(dropna=False))
    else:
        print(pd.crosstab(df[a], df[b], dropna=False))

def fit_lmer(formula, data, factors=None, REML=True):
    m = Lmer(formula, data=data)
    m.fit(factors=factors, REML=REML, summarize=False)
    return m

def safe_anova(model):
    try:
        return model.anova()
    except Exception as e:
        return f"anova() failed: {e}"

def safe_posthoc(model, marginal_vars, grouping_vars=None, p_adjust=P_ADJUST):
    try:
        return model.post_hoc(marginal_vars=marginal_vars, grouping_vars=grouping_vars, p_adjust=p_adjust)
    except Exception as e:
        return f"post_hoc failed: {e}"

def get_aic(model):
    try:
        return float(model.AIC)
    except Exception:
        return np.nan

def get_bic(model):
    try:
        return float(model.BIC)
```

```
    except Exception:
        return np.nan

def print_fit_stats(model, label):
    print(f"\n{label}")
    print(f"AIC: {get_aic(model):.3f}")
    print(f"BIC: {get_bic(model):.3f}")

def residual_diagnostics(model, prefix):
    try:
        residuals = np.asarray(model.residuals)
        fitted = np.asarray(model.fits)

        # residual vs fitted
        plt.figure(figsize=(6, 5))
        plt.scatter(fitted, residuals)
        plt.axhline(0, linestyle="--")
        plt.xlabel("Fitted values")
        plt.ylabel("Residuals")
        plt.title(f"{prefix}: Residuals vs Fitted")
        plt.tight_layout()
        plt.savefig(os.path.join(OUT_DIR, f"{prefix}_resid_vs_fitted.png"), dpi=300)
        plt.close()

        # QQ plot
        plt.figure(figsize=(6, 5))
        stats.probplot(residuals, dist="norm", plot=plt)
        plt.title(f"{prefix}: QQ plot")
        plt.tight_layout()
        plt.savefig(os.path.join(OUT_DIR, f"{prefix}_qqplot.png"), dpi=300)
        plt.close()

        return "diagnostic plots saved"
    except Exception as e:
        return f"diagnostic plotting failed: {e}"

def can_log(series):
    s = pd.to_numeric(series, errors="coerce").dropna()
    return len(s) > 0 and (s > 0).all()

def model_var_name(y):
    """Return the actual response column name to use in the model."""
    if FINAL_SCALE[y] == "log":
        return f"log_{y}"
    return y

# Load and clean
df_raw = pd.read_excel(FILE_PATH, sheet_name=SHEET_NAME)
df = clean(df_raw)

for orig, safe in SAFE_MAP.items():
    if orig in df.columns:
        df[safe] = df[orig]

# create final transformed columns according to chosen scale
for y in RESPONSES_SAFE:
    if FINAL_SCALE[y] == "log":
        if not can_log(df[y]):
            raise ValueError(f"{y} was chosen for log transformation, but non-positive values are
present.")
        df[f"log_{y}"] = np.log(df[y] + LOG_OFFSET)

print("\nDetected responses:")
print(RESPONSES_SAFE)

print("\nFinal transformation choices:")
for y in RESPONSES_SAFE:
    print(f"{y}: {FINAL_SCALE[y]}")

print("\nDistribution summary:")
for y in RESPONSES_SAFE:
    vals = df[y].dropna()
    print(f"{y}: N={len(vals)}, min={vals.min():.3f}, max={vals.max():.3f}, mean={vals.mean():.3f},
skew={vals.skew():.3f}")

# datasets
dAB = df.copy()
dAB["compartment_group"] = dAB[COL_COMP].apply(map_compartment_collapsed)
dAB = dAB.loc[dAB["compartment_group"].notna()].copy()

dA = dAB.copy()
dB = dAB.loc[dAB[COL_ROT].isin(ROT_LEVELS)].copy()

dC = df.copy()
dC = dC.loc[dC[COL_SPEC].eq("fomitopsis pinicola")].copy()
dC = dC.loc[dC[COL_ROT].eq("brown-rot")].copy()
dC = dC.loc[dC[COL_SUB].isin(SUB_LEVELS)].copy()
dC["compartment_group"] = dC[COL_COMP].apply(map_compartment_pinicola_fine)
```

```
dC = dC.loc[dC["compartment_group"].notna()].copy()

# MODEL A
print_header("MODEL A (RQ1): compartment (basidiocarp collapsed) + (1|Tree_ID)")

for y in RESPONSES_SAFE:
    y_model = model_var_name(y)
    need = [COL_TREE, "compartment_group", y_model]
    dd = dA[need].dropna().copy()

    if dd.shape[0] < 10 or dd["compartment_group"].nunique() < 2:
        print(f"\nSkipping {y}: too few data/levels.")
        continue

    factors = {"compartment_group": sorted(dd["compartment_group"].unique().tolist())}

    print("\n" + "-" * 110)
    print(f"Response: {y} | model variable: {y_model} | N={dd.shape[0]} | Trees={dd[COL_TREE].nunique()}")
    print("Cell counts:")
    cell_counts(dd, "compartment_group")

    f_full = f"{y_model} ~ compartment_group + (1|{COL_TREE})"
    f_red  = f"{y_model} ~ 1 + (1|{COL_TREE})"

    m_full_ml = fit_lmer(f_full, dd, factors=factors, REML=REML_LRT)
    m_red_ml  = fit_lmer(f_red,  dd, factors=factors, REML=REML_LRT)

    print_fit_stats(m_full_ml, "ML full model fit")
    print_fit_stats(m_red_ml, "ML reduced model fit")

    m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)
    print_fit_stats(m_full, "Final REML model fit")

    print("\nFinal REML fit coefficients:")
    print(m_full.coefs)

    print("\nANOVA (if available):")
    print(safe_anova(m_full))

    print("\nPost-hoc: compartment_group (pairwise)")
    print(safe_posthoc(m_full, marginal_vars="compartment_group"))

    print("\nDiagnostics:")
    print(residual_diagnostics(m_full, f"{y_model}_ModelA"))

# MODEL B
print_header("MODEL B (RQ2): compartment × rot-type (collapsed) + (1|Tree_ID)")

for y in RESPONSES_SAFE:
    y_model = model_var_name(y)
    need = [COL_TREE, "compartment_group", COL_ROT, y_model]
    dd = dB[need].dropna().copy()

    if dd.shape[0] < 10 or dd["compartment_group"].nunique() < 2 or dd[COL_ROT].nunique() < 2:
        print(f"\nSkipping {y}: too few data/levels.")
        continue

    factors = {
        "compartment_group": sorted(dd["compartment_group"].unique().tolist()),
        COL_ROT: ROT_LEVELS
    }

    print("\n" + "-" * 110)
    print(f"Response: {y} | model variable: {y_model} | N={dd.shape[0]} | Trees={dd[COL_TREE].nunique()}")
    print("Cell counts (compartment_group x Rot_type):")
    cell_counts(dd, "compartment_group", COL_ROT)

    f_full = f"{y_model} ~ compartment_group * {COL_ROT} + (1|{COL_TREE})"
    m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)

    print_fit_stats(m_full, "Final REML model fit")

    print("\nFinal REML fit coefficients:")
    print(m_full.coefs)

    print("\nANOVA (if available):")
    print(safe_anova(m_full))

    print("\nPost-hoc: Rot_type within each compartment_group")
    print(safe_posthoc(m_full, marginal_vars=COL_ROT, grouping_vars="compartment_group"))

    print("\nPost-hoc: compartment_group within each Rot_type")
    print(safe_posthoc(m_full, marginal_vars="compartment_group", grouping_vars=COL_ROT))

    print("\nDiagnostics:")
    print(residual_diagnostics(m_full, f"{y_model}_ModelB"))

# MODEL C
```

```
print_header("MODEL C (RQ3): compartment × substrate (F. pinicola; brown-rot) + (1|Tree_ID)")

for y in RESPONSES_SAFE:
    y_model = model_var_name(y)
    need = [COL_TREE, "compartment_group", COL_SUB, y_model]
    dd = dC[need].dropna().copy()

    if dd.shape[0] < 10 or dd["compartment_group"].nunique() < 2 or dd[COL_SUB].nunique() < 2:
        print(f"\nSkipping {y}: too few data/levels.")
        continue

    factors = {
        "compartment_group": sorted(dd["compartment_group"].unique().tolist()),
        COL_SUB: SUB_LEVELS
    }

    print("\n" + "-" * 110)
    print(f"Response: {y} | model variable: {y_model} | N={dd.shape[0]} | Trees={dd[COL_TREE].nunique()}")
    print("Cell counts (compartment_group x Substrate_type):")
    cell_counts(dd, "compartment_group", COL_SUB)

    f_full = f"{y_model} ~ compartment_group * {COL_SUB} + (1|{COL_TREE})"
    m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)

    print_fit_stats(m_full, "Final REML model fit")

    print("\nFinal REML fit coefficients:")
    print(m_full.coefs)

    print("\nANOVA (if available):")
    print(safe_anova(m_full))

    print("\nPost-hoc: Substrate_type within each compartment_group")
    print(safe_posthoc(m_full, marginal_vars=COL_SUB, grouping_vars="compartment_group"))

    print("\nPost-hoc: compartment_group within each Substrate_type")
    print(safe_posthoc(m_full, marginal_vars="compartment_group", grouping_vars=COL_SUB))

    print("\nDiagnostics:")
    print(residual_diagnostics(m_full, f"{y_model}_ModelC"))
```

```
# ============================================================================
# LMEM for organic compound AUC values using pymer4 (Lmer).
#
# Model A (RQ1): compartment effect
# Model B (RQ2): compartment × rot type
# Model C (RQ3): compartment × substrate type (F. pinicola only)
#
# Random structure: random intercept for Tree_ID
# Fixed effects tested using likelihood ratio tests (ML)
# Final estimates reported using REML
#
# Transformation choices:
# - All DRIFT compounds: raw
# ============================================================================

import re
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
from pymer4.models import Lmer

# file paths to fetch the data

FILE_PATH  = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\332B25LR_DRIFT\250807_Drift_Data_Ladina_Reich.xlsx"
SHEET_NAME = "AUC"

COL_SAMPLE = "Sample_ID"
COL_TREE   = "Tree_ID"
COL_ROT    = "Rot_type"
COL_SPEC   = "Species_type"
COL_SUB    = "Substrate_type"
COL_COMP   = "Compartment"

USE_COLS_META = [COL_SAMPLE, COL_TREE, COL_ROT, COL_SPEC, COL_SUB, COL_COMP]

# Original names in Excel
SAFE_MAP = {
    "Aliphatic_range / 2760 - 3040 cm-1": "Aliphatic",
    "Carboxylic_range / 1700 - 1765 cm-1": "Carboxylic",
    "Aromatic_CC_range_2 / 1600 - 1690 cm-1": "Aromatic_CC_2",
    "Lignin_range / 1485 - 1550 cm-1": "Lignin",
    "Polysaccharide_range / 1020 - 1080 cm-1": "Polysaccharide",
    "Aromatic_CC_range_1 / 1350 - 1395 cm-1": "Aromatic_CC_1",
    "Aromatic_CH_range1 / 761 - 806 cm-1": "Aromatic_CH_1",
```

```
    "Aromatic_CH_range2 / 806 - 830 cm-1": "Aromatic_CH_2",
    "Phenolic_cellulose_range / NULL": "Phenolic_cellulose"
}

ROT_LEVELS = ["white-rot", "brown-rot"]
SUB_LEVELS = ["hardwood", "softwood"]

ALPHA = 0.05
P_ADJUST = "holm"
REML_FINAL = True
REML_LRT   = False
LOG_OFFSET = 1e-6

# safe response names
RESPONSES_SAFE = [
    "Aliphatic",
    "Carboxylic",
    "Aromatic_CC_2",
    "Lignin",
    "Polysaccharide",
    "Aromatic_CC_1",
    "Aromatic_CH",
    "Phenolic_cellulose",
]

# final transformation choices
FINAL_SCALE = {
    "Aliphatic": "raw",
    "Carboxylic": "raw",
    "Aromatic_CC_2": "raw",
    "Lignin": "raw",
    "Polysaccharide": "raw",
    "Aromatic_CC_1": "raw",
    "Aromatic_CH": "log",
    "Phenolic_cellulose": "raw",
}

# cleaning
def to_num(x):
    if pd.isna(x):
        return np.nan
    s = str(x).strip().replace(",", ".")
    s = "".join(ch for ch in s if ch.isdigit() or ch in ".-+eE")
    return pd.to_numeric(s, errors="coerce")

def normalize_text(s):
    if pd.isna(s):
        return np.nan
    s = str(s).strip()
    s = s.replace("–", "-").replace("—", "-").replace("_", "-")
    s = re.sub(r"\s+", " ", s)
    return s.lower()

def clean(df: pd.DataFrame) -> pd.DataFrame:
    df = df.loc[:, ~df.columns.astype(str).str.startswith("Unnamed")]
    df = df.loc[:, ~df.columns.duplicated(keep="last")]

    keep = [c for c in USE_COLS_META if c in df.columns] + [c for c in SAFE_MAP if c in df.columns]
    df = df[keep].copy()

    for c in SAFE_MAP:
        if c in df.columns:
            df[c] = df[c].apply(to_num)

    if COL_TREE in df.columns:
        df[COL_TREE] = df[COL_TREE].astype(str).str.strip()
    if COL_SAMPLE in df.columns:
        df[COL_SAMPLE] = df[COL_SAMPLE].astype(str).str.strip()

    for c in [COL_ROT, COL_SUB, COL_COMP, COL_SPEC]:
        if c in df.columns:
            df[c] = df[c].apply(normalize_text)

    if COL_ROT in df.columns:
        df[COL_ROT] = df[COL_ROT].str.replace(" ", "-", regex=False).replace({
            "white rot": "white-rot",
            "brown rot": "brown-rot",
            "white-rot": "white-rot",
            "brown-rot": "brown-rot",
        })

    if COL_SUB in df.columns:
        df[COL_SUB] = df[COL_SUB].replace({
            "hard wood": "hardwood",
            "soft wood": "softwood"
        })

    if COL_SPEC in df.columns:
```

```
        df[COL_SPEC] = df[COL_SPEC].replace({
            "f. pinicola": "fomitopsis pinicola",
            "fomitopsis_pinicola": "fomitopsis pinicola",
        })

    return df

# mapping compartments
FRUITING_BODY_ALIASES = {
    "fruiting body",
    "fruiting body outside",
    "fruiting body inside",
    "fruiting body spores",
    "basidiocarp",
    "sporocarp",
}

WOOD_COMPARTMENTS = {
    "wood with mycelia",
    "wood without mycelia",
    "weathered wood",
    "bark",
}

def map_compartment_collapsed(s):
    if pd.isna(s):
        return np.nan
    s = normalize_text(s)

    if s in {
        "fruiting body",
        "fruiting body outside",
        "fruiting body inside",
        "fruiting body spores",
        "basidiocarp",
        "sporocarp",
    }:
        return "basidiocarp"

    if s in WOOD_COMPARTMENTS:
        return s

    return np.nan

def map_compartment_pinicola_fine(s):
    if pd.isna(s):
        return np.nan
    s = normalize_text(s)

    if s == "fruiting body outside":
        return "pileipellis"
    if s == "fruiting body inside":
        return "trama"
    if s == "fruiting body spores":
        return "hymenium"
    if s in {"fruiting body", "basidiocarp", "sporocarp"}:
        return "basidiocarp"

    if s in WOOD_COMPARTMENTS:
        return s

    return np.nan

# helpers
def print_header(title):
    print("\n" + "#" * 110)
    print(title)
    print("#" * 110)

def cell_counts(df, a, b=None):
    if b is None:
        print(df[a].value_counts(dropna=False))
    else:
        print(pd.crosstab(df[a], df[b], dropna=False))

def fit_lmer(formula, data, factors=None, REML=True):
    m = Lmer(formula, data=data)
    m.fit(factors=factors, REML=REML, summarize=False)
    return m

def safe_compare(full_model, reduced_model):
    try:
        return full_model.compare(reduced_model)
    except Exception as e:
        return f"compare() failed: {e}"

def safe_anova(model):
    try:
```

```
        return model.anova()
    except Exception as e:
        return f"anova() failed: {e}"

def safe_posthoc(model, marginal_vars, grouping_vars=None, p_adjust=P_ADJUST):
    try:
        return model.post_hoc(marginal_vars=marginal_vars, grouping_vars=grouping_vars, p_adjust=p_adjust)
    except Exception as e:
        return f"post_hoc failed: {e}"

def can_log(series):
    s = pd.to_numeric(series, errors="coerce").dropna()
    return len(s) > 0 and (s > 0).all()

def model_var_name(y):
    if FINAL_SCALE[y] == "log":
        return f"log_{y}"
    return y

# cleaning

df_raw = pd.read_excel(FILE_PATH, sheet_name=SHEET_NAME)

print("\n=== Raw columns in AUC sheet ===")
print(df_raw.columns.tolist())

df = clean(df_raw)

# create safe response columns
for orig, safe in SAFE_MAP.items():
    if orig in df.columns:
        df[safe] = df[orig]

# merge Aromatic_CH_1 and Aromatic_CH_2 into one variable
if {"Aromatic_CH_1", "Aromatic_CH_2"}.issubset(df.columns):
    df["Aromatic_CH"] = df[["Aromatic_CH_1", "Aromatic_CH_2"]].sum(axis=1, skipna=True)
    # if both are missing, keep NA instead of 0
    both_missing = df[["Aromatic_CH_1", "Aromatic_CH_2"]].isna().all(axis=1)
    df.loc[both_missing, "Aromatic_CH"] = np.nan
else:
    raise ValueError("Aromatic_CH_1 and/or Aromatic_CH_2 not found, cannot create merged Aromatic_CH.")

# create log-transformed column for merged Aromatic_CH
if FINAL_SCALE["Aromatic_CH"] == "log":
    if not can_log(df["Aromatic_CH"]):
        raise ValueError("Aromatic_CH was chosen for log transformation, but non-positive values are
present.")
    df["log_Aromatic_CH"] = np.log(df["Aromatic_CH"] + LOG_OFFSET)

print("\nDetected organic compound responses (AUC_sum excluded; Aromatic_CH merged):")
print(RESPONSES_SAFE)

print("\nFinal transformation choices:")
for y in RESPONSES_SAFE:
    print(f"{y}: {FINAL_SCALE[y]}")

print("\nRaw compartment counts:")
print(df[COL_COMP].value_counts(dropna=False))

# dataset
dAB = df.copy()
dAB["compartment_group"] = dAB[COL_COMP].apply(map_compartment_collapsed)
dAB = dAB.loc[dAB["compartment_group"].notna()].copy()

dA = dAB.copy()
dB = dAB.loc[dAB[COL_ROT].isin(ROT_LEVELS)].copy()

dC = df.copy()
dC = dC.loc[dC[COL_SPEC].eq("fomitopsis pinicola")].copy()
dC = dC.loc[dC[COL_ROT].eq("brown-rot")].copy()
dC = dC.loc[dC[COL_SUB].isin(SUB_LEVELS)].copy()
dC["compartment_group"] = dC[COL_COMP].apply(map_compartment_pinicola_fine)
dC = dC.loc[dC["compartment_group"].notna()].copy()

# check mapping
print("\n=== Check Model A/B compartment mapping ===")
print(pd.crosstab(dAB[COL_COMP], dAB["compartment_group"], dropna=False))

print("\n=== Unique compartments in Model A/B ===")
print(sorted(dAB["compartment_group"].dropna().unique()))

print("\n=== Check Model C compartment mapping ===")
print(pd.crosstab(dC[COL_COMP], dC["compartment_group"], dropna=False))

print("\n=== Unique compartments in Model C ===")
print(sorted(dC["compartment_group"].dropna().unique()))

# Model A
```

```
print_header("MODEL A (RQ1): compartment (basidiocarp collapsed) + (1|Tree_ID)")

for y in RESPONSES_SAFE:
    y_model = model_var_name(y)
    need = [COL_TREE, "compartment_group", y_model]
    dd = dA[need].dropna().copy()

    if dd.shape[0] < 10 or dd["compartment_group"].nunique() < 2:
        print(f"\nSkipping {y}: too few data/levels.")
        continue

    factors = {"compartment_group": sorted(dd["compartment_group"].unique().tolist())}

    print("\n" + "-" * 110)
    print(f"Response: {y} | model variable: {y_model} | N={dd.shape[0]} | Trees={dd[COL_TREE].nunique()}")
    print("Cell counts:")
    cell_counts(dd, "compartment_group")

    f_full = f"{y_model} ~ compartment_group + (1|{COL_TREE})"
    f_red  = f"{y_model} ~ 1 + (1|{COL_TREE})"

    m_full_ml = fit_lmer(f_full, dd, factors=factors, REML=REML_LRT)
    m_red_ml  = fit_lmer(f_red,  dd, factors=factors, REML=REML_LRT)

    print("\nLRT (drop compartment_group):")
    print(safe_compare(m_full_ml, m_red_ml))

    m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)
    print("\nFinal REML fit coefficients:")
    print(m_full.coefs)

    print("\nANOVA (if available):")
    print(safe_anova(m_full))

    print("\nPost-hoc: compartment_group (pairwise)")
    print(safe_posthoc(m_full, marginal_vars="compartment_group"))

# Model B
print_header("MODEL B (RQ2): compartment × rot-type (collapsed) + (1|Tree_ID)")

for y in RESPONSES_SAFE:
    y_model = model_var_name(y)
    need = [COL_TREE, "compartment_group", COL_ROT, y_model]
    dd = dB[need].dropna().copy()

    if dd.shape[0] < 10 or dd["compartment_group"].nunique() < 2 or dd[COL_ROT].nunique() < 2:
        print(f"\nSkipping {y}: too few data/levels.")
        continue

    factors = {
        "compartment_group": sorted(dd["compartment_group"].unique().tolist()),
        COL_ROT: ROT_LEVELS
    }

    print("\n" + "-" * 110)
    print(f"Response: {y} | model variable: {y_model} | N={dd.shape[0]} | Trees={dd[COL_TREE].nunique()}")
    print("Cell counts (compartment_group x Rot_type):")
    cell_counts(dd, "compartment_group", COL_ROT)

    f_full = f"{y_model} ~ compartment_group * {COL_ROT} + (1|{COL_TREE})"
    f_noA  = f"{y_model} ~ {COL_ROT} + (1|{COL_TREE})"
    f_noB  = f"{y_model} ~ compartment_group + (1|{COL_TREE})"
    f_noAB = f"{y_model} ~ compartment_group + {COL_ROT} + (1|{COL_TREE})"

    m_full_ml = fit_lmer(f_full, dd, factors=factors, REML=REML_LRT)
    m_noA_ml  = fit_lmer(f_noA,  dd, factors=factors, REML=REML_LRT)
    m_noB_ml  = fit_lmer(f_noB,  dd, factors=factors, REML=REML_LRT)
    m_noAB_ml = fit_lmer(f_noAB, dd, factors=factors, REML=REML_LRT)

    print("\nLRT (drop compartment_group):")
    print(safe_compare(m_full_ml, m_noA_ml))

    print("\nLRT (drop Rot_type):")
    print(safe_compare(m_full_ml, m_noB_ml))

    print("\nLRT (drop interaction):")
    print(safe_compare(m_full_ml, m_noAB_ml))

    m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)
    print("\nFinal REML fit coefficients:")
    print(m_full.coefs)

    print("\nANOVA (if available):")
    print(safe_anova(m_full))

    print("\nPost-hoc: Rot_type within each compartment_group")
    print(safe_posthoc(m_full, marginal_vars=COL_ROT, grouping_vars="compartment_group"))
```

```
    print("\nPost-hoc: compartment_group within each Rot_type")
    print(safe_posthoc(m_full, marginal_vars="compartment_group", grouping_vars=COL_ROT))

# Model C
print_header("MODEL C (RQ3): compartment × substrate type (F. pinicola only) + (1|Tree_ID)")

for y in RESPONSES_SAFE:
    y_model = model_var_name(y)
    need = [COL_TREE, "compartment_group", COL_SUB, y_model]
    dd = dC[need].dropna().copy()

    if dd.shape[0] < 10 or dd["compartment_group"].nunique() < 2 or dd[COL_SUB].nunique() < 2:
        print(f"\nSkipping {y}: too few data/levels.")
        continue

    factors = {
        "compartment_group": sorted(dd["compartment_group"].unique().tolist()),
        COL_SUB: SUB_LEVELS
    }

    print("\n" + "-" * 110)
    print(f"Response: {y} | model variable: {y_model} | N={dd.shape[0]} | Trees={dd[COL_TREE].nunique()}")
    print("Cell counts (compartment_group x Substrate_type):")
    cell_counts(dd, "compartment_group", COL_SUB)

    f_full = f"{y_model} ~ compartment_group * {COL_SUB} + (1|{COL_TREE})"
    f_noA  = f"{y_model} ~ {COL_SUB} + (1|{COL_TREE})"
    f_noB  = f"{y_model} ~ compartment_group + (1|{COL_TREE})"
    f_noAB = f"{y_model} ~ compartment_group + {COL_SUB} + (1|{COL_TREE})"

    m_full_ml = fit_lmer(f_full, dd, factors=factors, REML=REML_LRT)
    m_noA_ml  = fit_lmer(f_noA,  dd, factors=factors, REML=REML_LRT)
    m_noB_ml  = fit_lmer(f_noB,  dd, factors=factors, REML=REML_LRT)
    m_noAB_ml = fit_lmer(f_noAB, dd, factors=factors, REML=REML_LRT)

    print("\nLRT (drop compartment_group):")
    print(safe_compare(m_full_ml, m_noA_ml))

    print("\nLRT (drop Substrate_type):")
    print(safe_compare(m_full_ml, m_noB_ml))

    print("\nLRT (drop interaction):")
    print(safe_compare(m_full_ml, m_noAB_ml))

    m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)
    print("\nFinal REML fit coefficients:")
    print(m_full.coefs)

    print("\nANOVA (if available):")
    print(safe_anova(m_full))

    print("\nPost-hoc: Substrate_type within each compartment_group")
    print(safe_posthoc(m_full, marginal_vars=COL_SUB, grouping_vars="compartment_group"))

    print("\nPost-hoc: compartment_group within each Substrate_type")
    print(safe_posthoc(m_full, marginal_vars="compartment_group", grouping_vars=COL_SUB))
```

```
# ============================================================================
# Linear mixed-effects modelling for sterol concentrations using pymer4 (Lmer).
#
# Model A (RQ1): compartment effect
# Model C (RQ3): compartment × substrate type (F. pinicola only; Ergosterol only)
#
# Random structure: random intercept for Tree_ID
# Fixed effects tested using likelihood ratio tests (ML)
# Final estimates reported using REML
#
# Final transformation choices:
# - All sterols log-transformed
# - Non-positive values become NaN after log transform and are excluded later
# ============================================================================

import os
import re
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
from pymer4.models import Lmer

# file paths to fetch the data and store the tables
STEROL_FILE = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Calculations_sterols_Ladina_Reich.xlsx"
STEROL_SHEET = "sterol_results"
OUT_DIR = r"C:\Users\Ladina\OneDrive - Universität Zürich UZH\Master\Masterarbeit\sterol_model_output"
os.makedirs(OUT_DIR, exist_ok=True)

# column names
```

```
COL_SAMPLE = "Sample_ID"
COL_TREE   = "Tree_ID"
COL_SUB    = "Substrate_type"
COL_COMP   = "Compartment"
COL_SPEC   = "Species_type"
ERG_COL    = "Ergosterol"

OTHER_STEROL_COLS = [
    "Erg.2 (Dehydroergosterol)",
    "β-Sitosterol",
    "Stigmastanol",
    "γ-Sitostenone",
]

P_ADJUST   = "holm"
REML_FINAL = True
REML_LRT   = False
LOG_OFFSET = 1e-6
SUB_LEVELS = ["hardwood", "softwood"]

# helpers
def normalize_text(x):
    if pd.isna(x):
        return np.nan
    s = str(x).strip().lower()
    s = s.replace("−", "-").replace("_", " ")
    s = re.sub(r"\s+", " ", s)
    return s

def to_num(x):
    if pd.isna(x):
        return np.nan
    s = str(x).strip().replace(",", ".")
    s = "".join(ch for ch in s if ch.isdigit() or ch in ".-+eE")
    return pd.to_numeric(s, errors="coerce")

def safe_name(x):
    s = re.sub(r"[^A-Za-z0-9]+", "_", x).strip("_")
    if s and s[0].isdigit():
        s = "v_" + s
    return s

def print_header(title):
    print("\n" + "#" * 110)
    print(title)
    print("#" * 110)

def get_aic(model):
    try:
        return float(model.AIC)
    except Exception:
        return np.nan

def get_bic(model):
    try:
        return float(model.BIC)
    except Exception:
        return np.nan

def print_fit_stats(model, label):
    print(f"\n{label}")
    print(f"AIC: {get_aic(model):.3f}")
    print(f"BIC: {get_bic(model):.3f}")

def cell_counts(df, a, b=None):
    if b is None:
        print(df[a].value_counts(dropna=False))
    else:
        print(pd.crosstab(df[a], df[b], dropna=False))

def fit_lmer(formula, data, factors=None, REML=True):
    m = Lmer(formula, data=data)
    m.fit(factors=factors, REML=REML, summarize=False)
    return m

def safe_compare(full_model, reduced_model):
    try:
        return full_model.compare(reduced_model)
    except Exception as e:
        return f"compare() failed: {e}"

def safe_anova(model):
    try:
        return model.anova()
    except Exception as e:
        return f"anova() failed: {e}"

def safe_posthoc(model, marginal_vars, grouping_vars=None, p_adjust=P_ADJUST):
```

```
    try:
        return model.post_hoc(
            marginal_vars=marginal_vars,
            grouping_vars=grouping_vars,
            p_adjust=p_adjust
        )
    except Exception as e:
        return f"post_hoc failed: {e}"

def report_nonpositive(df, var, safe_var):
    vals = pd.to_numeric(df[safe_var], errors="coerce")
    n_bad = (vals <= 0).sum()
    min_val = vals.min(skipna=True)
    if n_bad > 0:
        print(f"Warning: {var} has {n_bad} non-positive value(s) (min={min_val}).")
        print("These will become NaN after log transformation and be excluded from the models.")

# for Model A: all fruiting-body tissues collapsed
def map_compartment_A(x):
    if pd.isna(x):
        return np.nan
    s = normalize_text(x)
    if s in {"fruiting body", "fruiting body outside", "fruiting body inside", "fruiting body spores"}:
        return "basidiocarp"
    if s in {"bark", "weathered wood", "wood with mycelia", "wood without mycelia"}:
        return s
    return np.nan

# for Model C: fruiting body tissues separated
def map_compartment_C(x):
    if pd.isna(x):
        return np.nan
    s = normalize_text(x)
    if s == "fruiting body outside":
        return "pileipellis"
    if s == "fruiting body inside":
        return "trama"
    if s == "fruiting body spores":
        return "hymenium"
    if s in {"bark", "weathered wood", "wood with mycelia", "wood without mycelia"}:
        return s
    return np.nan

# read in data
df = pd.read_excel(STEROL_FILE, sheet_name=STEROL_SHEET)

required_cols = [COL_TREE, COL_SUB, COL_COMP, ERG_COL]
missing = [c for c in required_cols if c not in df.columns]
if missing:
    raise KeyError(f"Missing required columns: {missing}")

present_other_sterols = [c for c in OTHER_STEROL_COLS if c in df.columns]
missing_other = [c for c in OTHER_STEROL_COLS if c not in df.columns]
if missing_other:
    print("Warning: these sterol columns were not found and will be skipped:")
    print(missing_other)

# cleaning
for c in [COL_SUB, COL_COMP]:
    df[c] = df[c].apply(normalize_text)

if COL_SPEC in df.columns:
    df[COL_SPEC] = df[COL_SPEC].apply(normalize_text)

df[COL_TREE] = df[COL_TREE].astype(str).str.strip()

df[COL_SUB] = df[COL_SUB].replace({
    "hard wood": "hardwood",
    "soft wood": "softwood",
    "hardwood": "hardwood",
    "softwood": "softwood"
})

df = df[df[COL_SUB].isin(SUB_LEVELS)].copy()

# cleaning sterol columns
sterol_vars = [ERG_COL] + present_other_sterols
for c in sterol_vars:
    df[c] = df[c].apply(to_num)

# creating safe names for models
manual_name_map = {
    "Ergosterol": "Ergosterol",
    "Erg.2 (Dehydroergosterol)": "Erg2_Dehydroergosterol",
    "β-Sitosterol": "beta_Sitosterol",
    "Stigmastanol": "Stigmastanol",
    "γ-Sitostenone": "gamma_Sitostenone",
}
```

```
safe_var_map = {}
for var in sterol_vars:
    if var in manual_name_map:
        safe = manual_name_map[var]
    else:
        safe = safe_name(var)
    df[safe] = df[var]
    safe_var_map[var] = safe

print("\nSafe sterol variable names:")
for k, v in safe_var_map.items():
    print(f"{k} -> {v}")

print("\nDistribution summary (raw):")
for var in sterol_vars:
    safe_var = safe_var_map[var]
    vals = df[safe_var].dropna()
    if len(vals) == 0:
        continue
    print(f"{var}: N={len(vals)}, min={vals.min():.6f}, max={vals.max():.6f}, mean={vals.mean():.6f},
skew={vals.skew():.3f}")


# log transforming all sterols
for var in sterol_vars:
    safe_var = safe_var_map[var]
    report_nonpositive(df, var, safe_var)
    df[f"log_{safe_var}"] = np.log(df[safe_var] + LOG_OFFSET)

print("\nLog-transformed sterol variables:")
for var in sterol_vars:
    safe_var = safe_var_map[var]
    print(f"{var} -> log_{safe_var}")

# creating model groupings
df["compartment_group_A"] = df[COL_COMP].apply(map_compartment_A)
df["compartment_group_C"] = df[COL_COMP].apply(map_compartment_C)

levels_A = [
    "bark",
    "basidiocarp",
    "weathered wood",
    "wood with mycelia",
    "wood without mycelia"
]

levels_C = [
    "bark",
    "hymenium",
    "pileipellis",
    "trama",
    "weathered wood",
    "wood with mycelia",
    "wood without mycelia"
]

# Model A
print_header("MODEL A (RQ1): compartment effect (all sterols, log-transformed) + (1|Tree_ID)")

for var in sterol_vars:
    safe_var = safe_var_map[var]
    y_model = f"log_{safe_var}"

    need = [COL_TREE, "compartment_group_A", y_model]
    dd = df[need].dropna().copy()
    dd = dd[dd["compartment_group_A"].isin(levels_A)].copy()

    if dd.shape[0] < 10 or dd["compartment_group_A"].nunique() < 2:
        print(f"\nSkipping {var}: too few data/levels.")
        continue

    dd["compartment_group_A"] = pd.Categorical(
        dd["compartment_group_A"],
        categories=levels_A,
        ordered=True
    )

    factors = {"compartment_group_A": levels_A}

    print("\n" + "-" * 110)
    print(f"Response: {var} | model variable: {y_model} | N={dd.shape[0]} |
Trees={dd[COL_TREE].nunique()}")
    print("Cell counts:")
    cell_counts(dd, "compartment_group_A")

    f_full = f"{y_model} ~ compartment_group_A + (1|{COL_TREE})"
    f_red  = f"{y_model} ~ 1 + (1|{COL_TREE})"
```

```
    m_full_ml = fit_lmer(f_full, dd, factors=factors, REML=REML_LRT)
    m_red_ml  = fit_lmer(f_red,  dd, factors=factors, REML=REML_LRT)

    print_fit_stats(m_full_ml, "ML full model fit")
    print_fit_stats(m_red_ml, "ML reduced model fit")

    print("\nLRT (drop compartment_group_A):")
    print(safe_compare(m_full_ml, m_red_ml))

    m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)
    print_fit_stats(m_full, "Final REML model fit")

    print("\nFinal REML fit coefficients:")
    print(m_full.coefs)

    print("\nANOVA (if available):")
    print(safe_anova(m_full))

    print("\nPost-hoc: compartment_group_A (pairwise)")
    print(safe_posthoc(m_full, marginal_vars="compartment_group_A"))

# Model C: Ergosterol only: compartment × substrate type (F. pinicola only)
print_header("MODEL C (RQ3): compartment × substrate type (Ergosterol, log-transformed; F. pinicola only) +
(1|Tree_ID)")

erg_safe = safe_var_map[ERG_COL]
y_model = f"log_{erg_safe}"

need_cols = [y_model, "compartment_group_C", COL_SUB, COL_TREE]
if COL_SPEC in df.columns:
    need_cols.append(COL_SPEC)

dd = df.dropna(subset=need_cols).copy()

if COL_SPEC in dd.columns:
    dd = dd[dd[COL_SPEC].astype(str).str.lower().str.contains("pinicola", na=False)].copy()

dd = dd[dd["compartment_group_C"].isin(levels_C)].copy()
dd = dd[dd[COL_SUB].isin(SUB_LEVELS)].copy()

if dd.empty:
    print("Skipping Model C: no usable rows.")
else:
    if dd["compartment_group_C"].nunique() < 2:
        print("Skipping Model C: fewer than 2 compartment levels.")
    elif dd[COL_SUB].nunique() < 2:
        print("Skipping Model C: fewer than 2 substrate levels.")
    else:
        dd["compartment_group_C"] = pd.Categorical(
            dd["compartment_group_C"],
            categories=levels_C,
            ordered=True
        )

        dd[COL_SUB] = pd.Categorical(
            dd[COL_SUB],
            categories=SUB_LEVELS,
            ordered=True
        )

        factors = {
            "compartment_group_C": levels_C,
            COL_SUB: SUB_LEVELS
        }

        print("\n" + "-" * 110)
        print(f"Response: {ERG_COL} | model variable: {y_model} | N={dd.shape[0]} |
Trees={dd[COL_TREE].nunique()}")
        print("Cell counts (compartment_group_C x Substrate_type):")
        cell_counts(dd, "compartment_group_C", COL_SUB)

        f_full = f"{y_model} ~ compartment_group_C * {COL_SUB} + (1|{COL_TREE})"
        f_noA  = f"{y_model} ~ {COL_SUB} + (1|{COL_TREE})"
        f_noB  = f"{y_model} ~ compartment_group_C + (1|{COL_TREE})"
        f_noAB = f"{y_model} ~ compartment_group_C + {COL_SUB} + (1|{COL_TREE})"

        m_full_ml = fit_lmer(f_full, dd, factors=factors, REML=REML_LRT)
        m_noA_ml  = fit_lmer(f_noA,  dd, factors=factors, REML=REML_LRT)
        m_noB_ml  = fit_lmer(f_noB,  dd, factors=factors, REML=REML_LRT)
        m_noAB_ml = fit_lmer(f_noAB, dd, factors=factors, REML=REML_LRT)

        print_fit_stats(m_full_ml, "ML full model fit")
        print_fit_stats(m_noA_ml, "ML reduced model fit (drop compartment_group_C)")
        print_fit_stats(m_noB_ml, "ML reduced model fit (drop Substrate_type)")
        print_fit_stats(m_noAB_ml, "ML reduced model fit (drop interaction)")

        print("\nLRT (drop compartment_group_C):")
```

```
        print(safe_compare(m_full_ml, m_noA_ml))

        print("\nLRT (drop Substrate_type):")
        print(safe_compare(m_full_ml, m_noB_ml))

        print("\nLRT (drop interaction):")
        print(safe_compare(m_full_ml, m_noAB_ml))

        m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)
        print_fit_stats(m_full, "Final REML model fit")

        print("\nFinal REML fit coefficients:")
        print(m_full.coefs)

        print("\nANOVA (if available):")
        print(safe_anova(m_full))

        print("\nPost-hoc: substrate type within each compartment (Holm)")
        print(
            safe_posthoc(
                m_full,
                marginal_vars=COL_SUB,
                grouping_vars="compartment_group_C",
                p_adjust="holm"
            )
        )

        print("\nPost-hoc: compartment within each substrate type (Holm)")
        print(
            safe_posthoc(
                m_full,
                marginal_vars="compartment_group_C",
                grouping_vars=COL_SUB,
                p_adjust="holm"
            )
        )
```

```
# ============================================================================
# Linear mixed-effects modelling for fatty acid variables using pymer4 (Lmer).
#
# Model A (RQ1): compartment effect
# Model C (RQ3): compartment × substrate type (F. pinicola only)
#
# Random structure: random intercept for Tree_ID
# Fixed effects tested using likelihood ratio tests (ML)
# Final estimates reported using REML
#
# Final transformation choices:
# - All selected FA variables log-transformed
# ============================================================================

import os
import re
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
from pymer4.models import Lmer

# file paths to fetch data and store model tables
FILE_PATH  = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Master_thesis_Ladina_Reich_excel_word_documents\Calculations_FA_Ladina_Reich.xlsx"
SHEET_NAME = "FA_results"
OUT_DIR = r"C:\Users\Ladina\OneDrive - Universität Zürich UZH\Master\Masterarbeit\FA_model_output"
os.makedirs(OUT_DIR, exist_ok=True)

# column names
COL_SAMPLE = "Sample_ID"
COL_TREE   = "Tree_ID"
COL_ROT    = "Rot_type"
COL_SPEC   = "Species_type"
COL_SUB    = "Substrate_type"
COL_COMP   = "Compartment"

P_ADJUST   = "holm"
REML_FINAL = True
REML_LRT   = False
LOG_OFFSET = 1e-6
SUB_LEVELS = ["hardwood", "softwood"]

# loading data
df = pd.read_excel(FILE_PATH, sheet_name=SHEET_NAME)

# cleaning text
def normalize_text(x):
    if pd.isna(x):
```

```
        return np.nan
    s = str(x).strip().lower()
    s = s.replace("−", "-").replace("–", "-").replace("_", " ")
    s = re.sub(r"\s+", " ", s)
    return s

for c in [COL_ROT, COL_SPEC, COL_SUB, COL_COMP]:
    if c in df.columns:
        df[c] = df[c].apply(normalize_text)

df[COL_SUB] = df[COL_SUB].replace({
    "hard wood": "hardwood",
    "soft wood": "softwood",
    "hardwood": "hardwood",
    "softwood": "softwood"
})

df[COL_TREE] = df[COL_TREE].astype(str).str.strip()

# compartment grouping
def map_compartment_A(x):
    if pd.isna(x):
        return np.nan
    x = normalize_text(x)
    if x in {"fruiting body", "fruiting body inside", "fruiting body outside", "fruiting body spores"}:
        return "basidiocarp"
    if x in {"bark", "weathered wood", "wood with mycelia", "wood without mycelia"}:
        return x
    return np.nan

def map_compartment_C(x):
    if pd.isna(x):
        return np.nan
    x = normalize_text(x)
    if x == "fruiting body outside":
        return "pileipellis"
    if x == "fruiting body inside":
        return "trama"
    if x == "fruiting body spores":
        return "hymenium"
    if x in {"bark", "weathered wood", "wood with mycelia", "wood without mycelia"}:
        return x
    return np.nan

df["compartment_group_A"] = df[COL_COMP].apply(map_compartment_A)
df["compartment_group_C"] = df[COL_COMP].apply(map_compartment_C)

# FA group
SFA = ["C14:0","C15:0","C16:0","C17:0","C18:0",
       "C20:0","C21:0","C22:0","C23:0","C24:0",
       "C25:0","C26:0","C28:0"]
UFA = ["C15:1","C16:1","C16:2","C18:1","C18:2",
       "C24:1","C25:1","C26:1"]
SCFA = ["C14:0","C15:0","C16:0","C17:0","C18:0",
        "C15:1","C16:1","C16:2","C18:1","C18:2"]
LCFA = ["C20:0","C21:0","C22:0","C23:0","C24:0",
        "C25:0","C26:0","C28:0","C24:1","C25:1","C26:1"]
BCFA = ["BCFA_17","BCFA_23","BCFA_25"]
DCA  = ["DCA_9","DCA_16","DCA_20","DCA_22"]
SSCFA = ["C14:0","C15:0","C16:0","C17:0","C18:0","C20:0"]
SLCFA = ["C21:0","C22:0","C23:0","C24:0","C25:0","C26:0","C28:0"]
USCFA = ["C15:1","C16:1","C16:2","C18:1","C18:2"]
ULCFA = ["C24:1","C25:1","C26:1"]

# numeric conversion
def to_num(x):
    if pd.isna(x):
        return np.nan
    s = str(x).strip().replace(",", ".")
    s = "".join(ch for ch in s if ch.isdigit() or ch in ".-+eE")
    return pd.to_numeric(s, errors="coerce")

fa_cols = sorted(set(
    SFA + UFA + SCFA + LCFA + BCFA + DCA + SSCFA + SLCFA + USCFA + ULCFA +
    ["C16:1", "C18:1", "C18:2", "C26:1"]
))

for c in fa_cols:
    if c in df.columns:
        df[c] = df[c].apply(to_num)

# calculating variables
def sum_fa(frame, cols):
    existing = [c for c in cols if c in frame.columns]
    if len(existing) == 0:
        return np.nan
    return frame[existing].sum(axis=1, skipna=True)
```

```
df["SFA"] = sum_fa(df, SFA)
df["UFA"] = sum_fa(df, UFA)
df["SCFA"] = sum_fa(df, SCFA)
df["LCFA"] = sum_fa(df, LCFA)
df["BCFA"] = sum_fa(df, BCFA)
df["DCA"] = sum_fa(df, DCA)
df["SSCFA"] = sum_fa(df, SSCFA)
df["SLCFA"] = sum_fa(df, SLCFA)
df["USCFA"] = sum_fa(df, USCFA)
df["ULCFA"] = sum_fa(df, ULCFA)

df["SFA_UFA_ratio"] = df["SFA"] / df["UFA"].replace(0, np.nan)
df["LCFA_SCFA_ratio"] = df["LCFA"] / df["SCFA"].replace(0, np.nan)

# variables to model
modelA_vars = [
    "SFA_UFA_ratio",
    "LCFA_SCFA_ratio",
    "BCFA",
    "DCA",
    "SSCFA",
    "SLCFA",
    "USCFA",
    "ULCFA"
]

modelC_vars = ["C16:1", "C18:1", "C18:2", "C26:1"]

all_vars = modelA_vars + modelC_vars

# log transformation
def safe_name(x):
    s = re.sub(r"[^A-Za-z0-9]+", "_", x).strip("_")
    if s and s[0].isdigit():
        s = "v_" + s
    return s

safe_var_map = {}
for var in all_vars:
    if var in df.columns:
        safe_var = safe_name(var)
        df[safe_var] = df[var]
        safe_var_map[var] = safe_var

print("\nSafe FA variable names:")
for k, v in safe_var_map.items():
    print(f"{k} -> {v}")

print("\nDistribution summary (raw):")
for var in all_vars:
    if var not in safe_var_map:
        continue
    safe_var = safe_var_map[var]
    vals = df[safe_var].dropna()
    if len(vals) == 0:
        continue
    print(f"{var}: N={len(vals)}, min={vals.min():.6f}, max={vals.max():.6f}, mean={vals.mean():.6f},
skew={vals.skew():.3f}")

# Log-transform all selected FA variables
for var in all_vars:
    if var not in safe_var_map:
        continue
    safe_var = safe_var_map[var]
    df[f"log_{safe_var}"] = np.log(df[safe_var] + LOG_OFFSET)

print("\nLog-transformed FA variables:")
for var in all_vars:
    if var in safe_var_map:
        print(f"{var} -> log_{safe_var_map[var]}")

levels_A = [
    "bark",
    "basidiocarp",
    "weathered wood",
    "wood with mycelia",
    "wood without mycelia"
]

levels_C = [
    "bark",
    "hymenium",
    "pileipellis",
    "trama",
    "weathered wood",
    "wood with mycelia",
    "wood without mycelia"
]
```

```
# helpers
def print_header(title):
    print("\n" + "#" * 110)
    print(title)
    print("#" * 110)

def get_aic(model):
    try:
        return float(model.AIC)
    except Exception:
        return np.nan

def get_bic(model):
    try:
        return float(model.BIC)
    except Exception:
        return np.nan

def print_fit_stats(model, label):
    print(f"\n{label}")
    print(f"AIC: {get_aic(model):.3f}")
    print(f"BIC: {get_bic(model):.3f}")

def cell_counts(df, a, b=None):
    if b is None:
        print(df[a].value_counts(dropna=False))
    else:
        print(pd.crosstab(df[a], df[b], dropna=False))

def fit_lmer(formula, data, factors=None, REML=True):
    m = Lmer(formula, data=data)
    m.fit(factors=factors, REML=REML, summarize=False)
    return m

def safe_compare(full_model, reduced_model):
    try:
        return full_model.compare(reduced_model)
    except Exception as e:
        return f"compare() failed: {e}"

def safe_anova(model):
    try:
        return model.anova()
    except Exception as e:
        return f"anova() failed: {e}"

def safe_posthoc(model, marginal_vars, grouping_vars=None, p_adjust=P_ADJUST):
    try:
        return model.post_hoc(
            marginal_vars=marginal_vars,
            grouping_vars=grouping_vars,
            p_adjust=p_adjust
        )
    except Exception as e:
        return f"post_hoc failed: {e}"

def is_pinicola(x):
    if pd.isna(x):
        return False
    s = str(x).strip().lower()
    return "pinicola" in s

# Model A
print_header("MODEL A (RQ1): compartment effect (all selected FA variables, log-transformed) +
(1|Tree_ID)")

for var in modelA_vars:
    if var not in safe_var_map:
        print(f"\nSkipping {var}: variable not found.")
        continue

    safe_var = safe_var_map[var]
    y_model = f"log_{safe_var}"

    need = [COL_TREE, "compartment_group_A", y_model]
    dd = df[need].dropna().copy()
    dd = dd[dd["compartment_group_A"].isin(levels_A)].copy()

    if dd.shape[0] < 10 or dd["compartment_group_A"].nunique() < 2:
        print(f"\nSkipping {var}: too few data/levels.")
        continue

    dd["compartment_group_A"] = pd.Categorical(
        dd["compartment_group_A"],
        categories=levels_A,
        ordered=True
    )
```

```
    factors = {"compartment_group_A": levels_A}

    print("\n" + "-" * 110)
    print(f"Response: {var} | model variable: {y_model} | N={dd.shape[0]} |
Trees={dd[COL_TREE].nunique()}")
    print("Cell counts:")
    cell_counts(dd, "compartment_group_A")

    f_full = f"{y_model} ~ compartment_group_A + (1|{COL_TREE})"
    f_red  = f"{y_model} ~ 1 + (1|{COL_TREE})"

    m_full_ml = fit_lmer(f_full, dd, factors=factors, REML=REML_LRT)
    m_red_ml  = fit_lmer(f_red,  dd, factors=factors, REML=REML_LRT)

    print_fit_stats(m_full_ml, "ML full model fit")
    print_fit_stats(m_red_ml, "ML reduced model fit")

    print("\nLRT (drop compartment_group_A):")
    print(safe_compare(m_full_ml, m_red_ml))

    m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)
    print_fit_stats(m_full, "Final REML model fit")

    print("\nFinal REML fit coefficients:")
    print(m_full.coefs)

    print("\nANOVA (if available):")
    print(safe_anova(m_full))

    print("\nPost-hoc: compartment_group_A (pairwise)")
    print(safe_posthoc(m_full, marginal_vars="compartment_group_A"))

# MODEL C
print_header("MODEL C (RQ3): compartment × substrate type (selected single FA variables, log-transformed;
F. pinicola only) + (1|Tree_ID)")

for var in modelC_vars:
    if var not in safe_var_map:
        print(f"\nSkipping {var}: variable not found.")
        continue

    safe_var = safe_var_map[var]
    y_model = f"log_{safe_var}"

    need = [COL_TREE, "compartment_group_C", COL_SUB, COL_SPEC, y_model]
    dd = df[need].dropna().copy()

    dd = dd[dd[COL_SPEC].apply(is_pinicola)].copy()
    dd = dd[dd["compartment_group_C"].isin(levels_C)].copy()
    dd = dd[dd[COL_SUB].isin(SUB_LEVELS)].copy()

    if dd.empty:
        print(f"\nSkipping {var}: no usable rows.")
        continue

    if dd["compartment_group_C"].nunique() < 2:
        print(f"\nSkipping {var}: fewer than 2 compartment levels.")
        continue

    if dd[COL_SUB].nunique() < 2:
        print(f"\nSkipping {var}: fewer than 2 substrate levels.")
        continue

    dd["compartment_group_C"] = pd.Categorical(
        dd["compartment_group_C"],
        categories=levels_C,
        ordered=True
    )

    dd[COL_SUB] = pd.Categorical(
        dd[COL_SUB],
        categories=SUB_LEVELS,
        ordered=True
    )

    factors = {
        "compartment_group_C": levels_C,
        COL_SUB: SUB_LEVELS
    }

    print("\n" + "-" * 110)
    print(f"Response: {var} | model variable: {y_model} | N={dd.shape[0]} |
Trees={dd[COL_TREE].nunique()}")
    print("Cell counts (compartment_group_C x Substrate_type):")
    cell_counts(dd, "compartment_group_C", COL_SUB)

    f_full = f"{y_model} ~ compartment_group_C * {COL_SUB} + (1|{COL_TREE})"
```

```
    f_noA  = f"{y_model} ~ {COL_SUB} + (1|{COL_TREE})"
    f_noB  = f"{y_model} ~ compartment_group_C + (1|{COL_TREE})"
    f_noAB = f"{y_model} ~ compartment_group_C + {COL_SUB} + (1|{COL_TREE})"

    m_full_ml = fit_lmer(f_full, dd, factors=factors, REML=REML_LRT)
    m_noA_ml  = fit_lmer(f_noA,  dd, factors=factors, REML=REML_LRT)
    m_noB_ml  = fit_lmer(f_noB,  dd, factors=factors, REML=REML_LRT)
    m_noAB_ml = fit_lmer(f_noAB, dd, factors=factors, REML=REML_LRT)

    print_fit_stats(m_full_ml, "ML full model fit")
    print_fit_stats(m_noA_ml, "ML reduced model fit (drop compartment_group_C)")
    print_fit_stats(m_noB_ml, "ML reduced model fit (drop Substrate_type)")
    print_fit_stats(m_noAB_ml, "ML reduced model fit (drop interaction)")

    print("\nLRT (drop compartment_group_C):")
    print(safe_compare(m_full_ml, m_noA_ml))

    print("\nLRT (drop Substrate_type):")
    print(safe_compare(m_full_ml, m_noB_ml))

    print("\nLRT (drop interaction):")
    print(safe_compare(m_full_ml, m_noAB_ml))

    m_full = fit_lmer(f_full, dd, factors=factors, REML=REML_FINAL)
    print_fit_stats(m_full, "Final REML model fit")

    print("\nFinal REML fit coefficients:")
    print(m_full.coefs)

    print("\nANOVA (if available):")
    print(safe_anova(m_full))

    print("\nPost-hoc: substrate type within each compartment (Holm)")
    print(
        safe_posthoc(
            m_full,
            marginal_vars=COL_SUB,
            grouping_vars="compartment_group_C",
            p_adjust="holm"
        )
    )

    print("\nPost-hoc: compartment within each substrate type (Holm)")
    print(
        safe_posthoc(
            m_full,
            marginal_vars="compartment_group_C",
            grouping_vars=COL_SUB,
            p_adjust="holm"
        )
    )
```

9.2.2 Mean values and figure visualization

```
#EA: Mean +- SD of TC and TN concentration and figure visualizations

# importing packages: numpy, pandas, matplotlib
import re
import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
from matplotlib.patches import Patch
from matplotlib.lines import Line2D

# file paths to fetch the data and store the figures and tables
FILE_PATH = r"C:\Users\Ladina\OneDrive - Universität Zürich UZH\Master\Masterarbeit\EA
data\Calculations_EA_Data_Ladina_Reich.xlsx"
SHEET_NAME = "EA_CN"
OUT_FIG_1 = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Figures\TC_TN_scatter_meanSD.png"
OUT_FIG_2 = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Figures\TC_TN_scatter_meanSD_rot_compartment.png"
OUT_FIG_3 = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Figures\brownrot_TC_TN_scatter_meanSD_substrate_compartment.png"
OUT_TABLE_1 = r"C:\Users\Ladina\OneDrive - Universität Zürich UZH\Master\Masterarbeit\EA
data\TC_TN_D13C_D15N_by_compartment.xlsx"
OUT_TABLE_2 = r"C:\Users\Ladina\OneDrive - Universität Zürich UZH\Master\Masterarbeit\EA
data\TC_TN_D13C_D15N_by_compartment_and_rot.xlsx"
OUT_TABLE_3 = r"C:\Users\Ladina\OneDrive - Universität Zürich UZH\Master\Masterarbeit\EA
data\TC_TN_D13C_D15N_brownrot_by_compartment_and_substrate.xlsx"

# figure style
STYLE = {
```

```
    "font_family": "sans-serif", "font_sans": ["Arial"], "ylabel_fontsize": 14, "xlabel_fontsize":
14,"xtick_fontsize": 14,
    "ytick_fontsize": 14,"legend_fontsize": 14, "legend_title_fontsize": 15,"label_color":
"black","point_alpha": 0.55,
    "point_size": 35, "point_edge_color": "black", "point_edge_lw": 0.4, "mean_marker":
"D","mean_marker_size": 12,
    "mean_size": 95,"mean_edge_color": "black", "mean_edge_lw": 1.0,"sd_lw": 1.8, "sd_capsize":
4,"sd_alpha": 0.95, "legend_frameon": True,
}

# coloring of figure
COMPARTMENT_COLOR_MAP = {
    "basidiocarp": "#1f4e79",
    "wood with mycelia": "#6baed6",
    "wood without mycelia": "#e6550d",
    "weathered wood": "#fd8d3c",
    "bark": "#9467bd",
}

ROT_COLOR_MAP = {"white-rot": "#1f77b4", "brown-rot": "#8c564b"}

SUBSTRATE_COLOR_MAP = {"hardwood": "#C9C28E", "softwood": "#8B5A2B"}

COMPARTMENT_MARKER_MAP = {
    "basidiocarp": "o",
    "pileipellis": "o",
    "trama": "v",
    "hymenium": "X",
    "wood with mycelia": "s",
    "wood without mycelia": "^",
    "weathered wood": "D",
    "bark": "P",
}

# column names of the excel files
COL_TREE = "Tree_ID"
COL_COMP = "Compartment"
COL_TC = "TC concentration [%]"
COL_TN = "TN concentration [%]"
COL_D13C = "Delta13C [‰]"
COL_D15N = "Delta15N [‰]"
COL_RT = "Rot_type"
COL_SUB = "Substrate_type"

# cleaning the data
def to_num(x):
    if pd.isna(x):
        return np.nan
    s = str(x).strip().replace(",", ".")
    s = "".join(ch for ch in s if ch.isdigit() or ch in ".-+eE")
    return pd.to_numeric(s, errors="coerce")

def normalize_text(s):
    if pd.isna(s):
        return np.nan
    s = str(s).strip()
    s = s.replace("–", "-").replace("—", "-").replace("_", "-")
    s = re.sub(r"\s+", " ", s)
    return s.lower()

def clean_dataframe(df):
    df = df.loc[:, ~df.columns.astype(str).str.startswith("Unnamed")]
    df = df.loc[:, ~df.columns.duplicated(keep="last")]
    for c in [COL_TC, COL_TN, COL_D13C, COL_D15N]:
        if c in df.columns:
            df[c] = df[c].apply(to_num)
    for c in [COL_COMP, COL_RT, COL_SUB]:
        if c in df.columns:
            df[c] = df[c].apply(normalize_text)
    return df

# compartment mapping for different research questions
FRUITING_BODY_ALIASES = {
    "fruiting body", "fruiting body outside", "fruiting body inside", "fruiting body spores",
"basidiocarp", "sporocarp",
}
WOOD_COMPARTMENTS = {"wood with mycelia", "wood without mycelia", "weathered wood", "bark"}

def map_compartment_collapsed(s):
    if pd.isna(s):
        return np.nan
    s = normalize_text(s)
    if s in FRUITING_BODY_ALIASES or s.startswith("fruiting body"):
        return "basidiocarp"
    if s in WOOD_COMPARTMENTS:
        return s
    return np.nan
```

```python
def map_compartment_detailed(s):
    if pd.isna(s):
        return np.nan
    s = str(s).strip().lower()
    if s == "fruiting body outside": return "pileipellis"
    if s == "fruiting body inside": return "trama"
    if s == "fruiting body spores": return "hymenium"
    if s in WOOD_COMPARTMENTS: return s
    if s == "basidiocarp": return "basidiocarp"
    return np.nan

# summary table (TC, TN, Δ13C, Δ15N)
def create_summary_table(df, group_cols):
    cols = [COL_TC, COL_TN, COL_D13C, COL_D15N]
    for c in cols:
        if c in df.columns:
            df[c] = pd.to_numeric(df[c].astype(str).str.replace(",", ".", regex=False), errors="coerce")

    summary = (
        df.groupby(group_cols)
        .agg(
            TC_mean=(COL_TC, "mean"), TC_sd=(COL_TC, "std"), TC_n=(COL_TC, "count"),
            TN_mean=(COL_TN, "mean"), TN_sd=(COL_TN, "std"), TN_n=(COL_TN, "count"),
            D13C_mean=(COL_D13C, "mean"), D13C_sd=(COL_D13C, "std"), D13C_n=(COL_D13C, "count"),
            D15N_mean=(COL_D15N, "mean"), D15N_sd=(COL_D15N, "std"), D15N_n=(COL_D15N, "count"),
        )
        .reset_index()
    )

    for col in ["TC_sd","TN_sd","D13C_sd","D15N_sd"]:
        summary[col] = summary[col].fillna(0)

    def fmt(prefix):
        return (
            summary[f"{prefix}_mean"].round(2).astype(str)
            + " ± " + summary[f"{prefix}_sd"].round(2).astype(str)
            + " (n=" + summary[f"{prefix}_n"].astype(int).astype(str) + ")"
        )

    summary["TC"] = fmt("TC")
    summary["TN"] = fmt("TN")
    summary["Delta13C"] = fmt("D13C")
    summary["Delta15N"] = fmt("D15N")

    return summary[group_cols + ["TC", "TN", "Delta13C", "Delta15N"]]

# Figure 1: TC vs. TN by compartment
def plot_figure_1(df, target_order, out_fig=None, out_table=None):
    dd = df.dropna(subset=["compartment_group", COL_TC, COL_TN]).copy()
    dd = dd[dd["compartment_group"].isin(target_order)].copy()

    fig, ax = plt.subplots(figsize=(12, 8), dpi=600)
    for comp in target_order:
        sub = dd.loc[dd["compartment_group"] == comp, [COL_TC, COL_TN]].dropna()
        if sub.empty: continue
        x = sub[COL_TC].to_numpy(); y = sub[COL_TN].to_numpy()
        ax.scatter(x, y, s=STYLE["point_size"], alpha=STYLE["point_alpha"],
                   c=COMPARTMENT_COLOR_MAP.get(comp, "#cccccc"),
                   edgecolors=STYLE["point_edge_color"], linewidths=STYLE["point_edge_lw"])
        x_mean, y_mean = float(np.nanmean(x)), float(np.nanmean(y))
        x_sd, y_sd = float(np.nanstd(x, ddof=1)), float(np.nanstd(y, ddof=1))
        ax.errorbar(x_mean, y_mean, xerr=x_sd, yerr=y_sd, fmt=STYLE["mean_marker"],
                    markersize=np.sqrt(STYLE["mean_size"]),
                    mfc=COMPARTMENT_COLOR_MAP.get(comp, "#cccccc"),
                    mec=STYLE["mean_edge_color"], mew=STYLE["mean_edge_lw"],
                    ecolor=COMPARTMENT_COLOR_MAP.get(comp, "#cccccc"),
                    elinewidth=STYLE["sd_lw"], capsize=STYLE["sd_capsize"],
                    alpha=STYLE["sd_alpha"], zorder=5)
    ax.set_xlabel(COL_TC, fontsize=STYLE["xlabel_fontsize"], color=STYLE["label_color"])
    ax.set_ylabel(COL_TN, fontsize=STYLE["ylabel_fontsize"], color=STYLE["label_color"])
    ax.tick_params(axis="x", labelsize=STYLE["xtick_fontsize"])
    ax.tick_params(axis="y", labelsize=STYLE["ytick_fontsize"])
    ax.spines["top"].set_visible(True)
    ax.spines["right"].set_visible(True)
    handles = [Patch(facecolor=COMPARTMENT_COLOR_MAP.get(c, "#cccccc"), edgecolor="black", label=c) for c
in target_order]
    ax.legend(handles=handles, title="Compartment", fontsize=STYLE["legend_fontsize"],
              title_fontsize=STYLE["legend_title_fontsize"], frameon=True, loc="upper right")
    plt.tight_layout()
    if out_fig: plt.savefig(out_fig, dpi=600, bbox_inches="tight")
    plt.show()

    summary_fmt = create_summary_table(dd, ["compartment_group"])
    print("\n=== FIGURE 1: Summary Table ===\n", summary_fmt)
    if out_table: summary_fmt.to_excel(out_table, index=False)

# Figure 2: TC vs. TN by rot type and compartment
def plot_figure_2(df, target_order, out_fig=None, out_table=None):
```

```
"""Scatter plot: color = rot type, marker = compartment with two legends."""
rots = ["white-rot", "brown-rot"]

sub = df[
    (df["compartment_group"].isin(target_order)) &
    (df[COL_RT].isin(rots))
].dropna(subset=[COL_TC, COL_TN])

fig, ax = plt.subplots(1, 1, figsize=(12, 8), dpi=400)

# --- scatter and mean±SD ---
for comp in target_order:
    for rot in rots:
        ss = sub[(sub["compartment_group"] == comp) & (sub[COL_RT] == rot)]
        if ss.empty:
            continue

        x = ss[COL_TC].to_numpy()
        y = ss[COL_TN].to_numpy()

        ax.scatter(
            x, y,
            s=STYLE["point_size"],
            alpha=STYLE["point_alpha"],
            c=ROT_COLOR_MAP[rot],
            marker=COMPARTMENT_MARKER_MAP.get(comp, "o"),
            edgecolors=STYLE["point_edge_color"],
            linewidths=STYLE["point_edge_lw"],
            zorder=2
        )

        # mean ± SD
        x_mean = np.nanmean(x)
        y_mean = np.nanmean(y)
        x_sd = np.nanstd(x, ddof=1) if len(x) > 1 else 0.0
        y_sd = np.nanstd(y, ddof=1) if len(y) > 1 else 0.0

        ax.errorbar(
            x_mean, y_mean,
            xerr=x_sd, yerr=y_sd,
            fmt=COMPARTMENT_MARKER_MAP.get(comp, "o"),
            markersize=STYLE["mean_marker_size"],
            mfc=ROT_COLOR_MAP[rot],
            mec="black",
            mew=0.9,
            ecolor=ROT_COLOR_MAP[rot],
            elinewidth=STYLE["sd_lw"],
            capsize=STYLE["sd_capsize"],
            alpha=STYLE["sd_alpha"],
            zorder=5,
        )

# === axis formatting ===
ax.set_xlabel(COL_TC, fontsize=STYLE["xlabel_fontsize"])
ax.set_ylabel(COL_TN, fontsize=STYLE["ylabel_fontsize"])
ax.tick_params(axis="both", labelsize=STYLE["xtick_fontsize"])
ax.spines["top"].set_visible(True)
ax.spines["right"].set_visible(True)

# === Legend 1: Rot type (color legend) ===
rot_handles = [
    Patch(facecolor=ROT_COLOR_MAP["white-rot"], edgecolor="black", label="White rot"),
    Patch(facecolor=ROT_COLOR_MAP["brown-rot"], edgecolor="black", label="Brown rot"),
]
leg_rot = ax.legend(
    handles=rot_handles,
    title="Rot type",
    loc="upper right",
    bbox_to_anchor=(1.0, 0.75),
    frameon=STYLE["legend_frameon"],
    fontsize=STYLE["legend_fontsize"],
    title_fontsize=STYLE["legend_title_fontsize"],
    fancybox=True,
)
ax.add_artist(leg_rot)  # keep when adding second legend

# === Legend 2: Compartment (symbol legend) ===
comp_handles = [
    Line2D(
        [0], [0],
        marker=COMPARTMENT_MARKER_MAP.get(comp, "o"),
        linestyle="",
        markerfacecolor="black",
        markeredgecolor="black",
        markersize=9,
        label=comp,
    )
    for comp in target_order
```

```
    ]
    ax.legend(
        handles=comp_handles,
        title="Compartment",
        loc="upper right",
        bbox_to_anchor=(1.0, 1.0),
        frameon=STYLE["legend_frameon"],
        fancybox=True,
        fontsize=STYLE["legend_fontsize"],
        title_fontsize=STYLE["legend_title_fontsize"],
    )

    plt.tight_layout()

    if out_fig:
        plt.savefig(out_fig, dpi=600, bbox_inches="tight")
        print(f"Figure 2 saved: {out_fig}")

    plt.show()

    # === Extended summary table (TC,TN,Δ13C,Δ15N) ===
    summary_fmt = create_summary_table(sub, ["compartment_group", COL_RT])
    print("\n=== FIGURE 2: Summary Table ===")
    print(summary_fmt.to_string(index=False))
    if out_table:
        summary_fmt.to_excel(out_table, index=False)
        print(f"Table saved: {out_table}")

    return summary_fmt

# Figure 3: Brown rot only, by substrate and compartment(basidiocarp splitted)
def plot_figure_3(df, target_order, out_fig=None, out_table=None):
    """Brown-rot only: color = substrate type, marker = compartment (detailed)."""
    subs = ["hardwood", "softwood"]

    # Filter data
    sub = df[
        (df["compartment_group"].isin(target_order)) &
        (df[COL_SUB].isin(subs)) &
        (df[COL_RT] == "brown-rot")
    ].dropna(subset=[COL_TC, COL_TN])

    fig, ax = plt.subplots(1, 1, figsize=(12, 8), dpi=500)

    # === Points and mean ± SD ===
    for comp in target_order:
        for sub_type in subs:
            ss = sub[(sub["compartment_group"] == comp) & (sub[COL_SUB] == sub_type)]
            if ss.empty:
                continue

            x = ss[COL_TC].to_numpy()
            y = ss[COL_TN].to_numpy()

            # --- scatter points colored by substrate ---
            ax.scatter(
                x, y,
                s=STYLE["point_size"],
                alpha=STYLE["point_alpha"],
                color=SUBSTRATE_COLOR_MAP[sub_type],
                marker=COMPARTMENT_MARKER_MAP.get(comp, "o"),
                edgecolor=STYLE["point_edge_color"],
                linewidth=STYLE["point_edge_lw"],
                zorder=2
            )

            # --- mean ± SD (same color as substrate) ---
            xm = float(np.nanmean(x))
            ym = float(np.nanmean(y))
            xsd = float(np.nanstd(x, ddof=1)) if len(x) > 1 else 0.0
            ysd = float(np.nanstd(y, ddof=1)) if len(y) > 1 else 0.0

            ax.errorbar(
                xm, ym,
                xerr=xsd, yerr=ysd,
                fmt=COMPARTMENT_MARKER_MAP.get(comp, "o"),
                markersize=STYLE["mean_marker_size"],
                mfc=SUBSTRATE_COLOR_MAP[sub_type],   # filled with substrate color
                mec="black",
                mew=1.0,
                ecolor=SUBSTRATE_COLOR_MAP[sub_type],
                elinewidth=STYLE["sd_lw"],
                capsize=STYLE["sd_capsize"],
                alpha=STYLE["sd_alpha"],
                zorder=5
            )

    # === Axis & formatting ===
```

```
    ax.set_xlabel(COL_TC, fontsize=STYLE["xlabel_fontsize"])
    ax.set_ylabel(COL_TN, fontsize=STYLE["ylabel_fontsize"])
    ax.tick_params(axis="x", labelsize=STYLE["xtick_fontsize"])
    ax.tick_params(axis="y", labelsize=STYLE["ytick_fontsize"])
    ax.spines["top"].set_visible(True)
    ax.spines["right"].set_visible(True)

    # === Legends ===
    # 1) Substrate (color legend)
    sub_handles = [
        Patch(facecolor=SUBSTRATE_COLOR_MAP["hardwood"], edgecolor="black", label="Hardwood"),
        Patch(facecolor=SUBSTRATE_COLOR_MAP["softwood"], edgecolor="black", label="Softwood"),
    ]
    leg_sub = ax.legend(
        handles=sub_handles,
        title="Substrate type",
        loc="upper right",
        bbox_to_anchor=(1.0, 0.66),
        frameon=STYLE["legend_frameon"],
        fancybox=True,
        fontsize=STYLE["legend_fontsize"],
        title_fontsize=STYLE["legend_title_fontsize"],
    )
    ax.add_artist(leg_sub)

    # 2) Compartment (symbol legend)
    comp_handles = [
        Line2D([0], [0],
               marker=COMPARTMENT_MARKER_MAP.get(comp, "o"),
               linestyle="", markerfacecolor="black",
               markeredgecolor="black", markersize=9, label=comp)
        for comp in target_order
    ]
    ax.legend(
        handles=comp_handles,
        title="Compartment",
        loc="upper right",
        bbox_to_anchor=(1.0, 1.0),
        frameon=STYLE["legend_frameon"],
        fancybox=True,
        fontsize=STYLE["legend_fontsize"],
        title_fontsize=STYLE["legend_title_fontsize"],
    )

    plt.tight_layout()
    if out_fig:
        plt.savefig(out_fig, dpi=600, bbox_inches="tight")
        print(f"Figure 3 saved: {out_fig}")

    plt.show()

    # === Summary table with δ13C & δ15N ===
    summary_fmt = create_summary_table(sub, ["compartment_group", COL_SUB])
    print("\n=== FIGURE 3: Summary Table ===")
    print(summary_fmt.to_string(index=False))
    if out_table:
        summary_fmt.to_excel(out_table, index=False)
        print(f"Table saved: {out_table}")

    return summary_fmt

# printing results
if __name__ == "__main__":
    df = pd.read_excel(FILE_PATH, sheet_name=SHEET_NAME)
    df = clean_dataframe(df)
    df["compartment_group_collapsed"] = df[COL_COMP].apply(map_compartment_collapsed)
    df["compartment_group_detailed"]  = df[COL_COMP].apply(map_compartment_detailed)

    TARGET_COLLAPSED = ["basidiocarp","wood with mycelia","wood without mycelia","weathered wood","bark"]
    TARGET_DETAILED  = ["pileipellis","trama","hymenium","wood with mycelia","wood without
mycelia","weathered wood","bark"]

    print("="*60,"\nFIGURE 1: TC vs TN by Compartment","\n","="*60)
    plot_figure_1(df.rename(columns={"compartment_group_collapsed":"compartment_group"}),
                  TARGET_COLLAPSED, OUT_FIG_1, OUT_TABLE_1)

    print("="*60,"\nFIGURE 2: TC vs TN by Rot Type and Compartment","\n","="*60)
    plot_figure_2(df.rename(columns={"compartment_group_collapsed":"compartment_group"}),
                  TARGET_COLLAPSED, OUT_FIG_2, OUT_TABLE_2)

    print("="*60,"\nFIGURE 3: Brown-rot TC vs TN by Substrate and Compartment","\n","="*60)
    plot_figure_3(df.rename(columns={"compartment_group_detailed":"compartment_group"}),
                  TARGET_DETAILED, OUT_FIG_3, OUT_TABLE_3)
```

#EA: figure visualization of δ¹³C and δ¹⁵N values

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.lines import Line2D
from matplotlib.patches import Ellipse
from scipy.stats import chi2
import unicodedata
import re

# file paths to fetch the data
FILE_PATH = r"C:\Users\Ladina\OneDrive - Universität Zürich UZH\Master\Masterarbeit\EA
data\Calculations_EA_Data_Ladina_Reich.xlsx"
SHEET_NAME = "EA_CN"

# figure style
STYLE = {
    "font_family": "sans-serif","font_sans": ["Arial"],"xlabel_fontsize": 15,"ylabel_fontsize":
15,"xtick_fontsize": 13,"ytick_fontsize": 13,"legend_fontsize": 13,"legend_title_fontsize":
13,"legend_frameon": True,"label_color": "black",
}

# coloring and marker setting
COMPARTMENT_MARKERS = {
    "pileipellis": "o",
    "trama": "P",
    "hymenium": "X",
    "fruiting body": "o",
    "wood with mycelia": "^",
    "wood without mycelia": "v",
    "weathered wood": "D",
    "bark": "s",
}

def grouped_palette(custom_palette=None):
    base = {
        "pileipellis": "#6b79d6",
        "trama": "#377eb8",
        "hymenium": "#69c3a3",
        "fruiting body": "#1f4e79",
        "wood with mycelia": "#6baed6",
        "wood without mycelia": "#e6550d",
        "weathered wood": "#fd8d3c",
        "bark": "#9467bd",
    }
    if custom_palette:
        base.update(custom_palette)
    return base

# column names of the excel files
COL_SPECIES = "Species_type"
COL_SUB     = "Substrate_type"
COL_COMP    = "Compartment"
COL_RT      = "Rot_type"
COL_C13     = "Delta13C [‰]"
COL_N15     = "Delta15N [‰]"

# cleaning the data
def clean_header(s):
    s = unicodedata.normalize("NFKC", str(s))
    s = s.replace("\u00a0", " ").replace("\u200b", "").replace("\ufeff", "")
    return s.strip()

def clean_text(x):
    if pd.isna(x):
        return np.nan
    s = unicodedata.normalize("NFKC", str(x))
    return s.replace("\u00a0", " ").replace("\u200b", "").strip()

def to_numeric_safe(series):
    s = series.astype(str).str.strip().str.replace(",", ".", regex=False)
    s = s.str.replace(r"[^\d\.\-\+eE]", "", regex=True)
    return pd.to_numeric(s, errors="coerce")

# confidence intervall as ellipse
def confidence_ellipse(x, y, ax, level=0.95, edgecolor="black", lw=1.2, alpha=0.18):
    x, y = np.asarray(x, float), np.asarray(y, float)
    ok = np.isfinite(x) & np.isfinite(y)
    x, y = x[ok], y[ok]
    if len(x) < 3:
        return None
    cov = np.cov(np.vstack([x, y]), ddof=1)
    eigvals, eigvecs = np.linalg.eigh(cov)
    if np.any(eigvals <= 0):
        return None
    k2 = chi2.ppf(level, 2)
    width, height = 2.0 * np.sqrt(eigvals * k2)
    angle = np.degrees(np.arctan2(eigvecs[1, 0], eigvecs[0, 0]))
```

```
    mean = [x.mean(), y.mean()]
    ell = Ellipse(mean, width, height, angle=angle,
                  edgecolor=edgecolor, facecolor="none", lw=lw, alpha=alpha)
    ax.add_patch(ell)
    return ell

# standardizing data
df2 = df.copy()
df2.columns = [clean_header(c) for c in df2.columns]
for c in [COL_SPECIES, COL_SUB, COL_COMP, COL_RT]:
    df2[c] = df2[c].map(clean_text)

df2[COL_C13] = to_numeric_safe(df2[COL_C13])
df2[COL_N15] = to_numeric_safe(df2[COL_N15])

for c in [COL_SPECIES, COL_SUB, COL_COMP, COL_RT]:
    df2[c] = df2[c].astype(str).str.strip().str.lower()

df2.loc[df2[COL_SPECIES].isin(["trametes versicolor", "trametes sp.", "trametes spp."]), COL_SPECIES] =
"trametes"
df2[COL_SPECIES] = df2[COL_SPECIES].replace({r"^formitopsis": "fomitopsis"}, regex=True)

# filter
def filter_isotope_data(df_in, species, substrate=None, split_fruiting=False):
    d = df_in.copy()

    # listify
    def as_list(x):
        if x is None:
            return None
        if isinstance(x, (list, tuple, set, np.ndarray, pd.Series)):
            return [str(i).strip().lower() for i in x]
        return [str(x).strip().lower()]
    species_list = as_list(species)
    sub_list = as_list(substrate)

    mask = pd.Series(True, index=d.index)
    if species_list is not None:
        pat = "|".join([re.escape(s) for s in species_list])
        mask &= d[COL_SPECIES].str.contains(pat, case=False, na=False)
    if sub_list is not None:
        mask &= d[COL_SUB].isin(sub_list)

    d = d.loc[mask, [COL_SPECIES, COL_SUB, COL_COMP, COL_C13, COL_N15]].dropna()

    # map compartments
    comp = d[COL_COMP].astype(str).str.strip().str.lower()
    if split_fruiting:
        comp_map = {
            "fruiting body outside": "pileipellis",
            "fruiting body inside": "trama",
            "fruiting body spores": "hymenium",
        }
    else:
        comp_map = {
            "fruiting body outside": "fruiting body",
            "fruiting body inside": "fruiting body",
            "fruiting body spores": "fruiting body",
        }
    d[COL_COMP] = comp.replace(comp_map)
    return d

# panel plotting
def plot_isotopes_on_ax(df_in, species, substrate, ax,
                        split_fruiting=False, palette=None, title=None):
    if palette is None:
        palette = grouped_palette()
    d = filter_isotope_data(df_in, species=species, substrate=substrate, split_fruiting=split_fruiting)
    if d.empty:
        ax.text(0.5, 0.5, "No data", ha="center", va="center")
        if title:
            ax.set_title(title, fontsize=STYLE["ylabel_fontsize"])
        return

    for comp, subdf in d.groupby(COL_COMP):
        x, y = subdf[COL_C13].to_numpy(), subdf[COL_N15].to_numpy()
        n = len(x)
        col = palette.get(comp, "#555")
        mark = COMPARTMENT_MARKERS.get(comp, "o")
        ax.scatter(x, y, color=col, marker=mark, s=50, alpha=0.4)
        if n >= 2:
            mx, my = np.mean(x), np.mean(y)
            sx, sy = np.std(x, ddof=1), np.std(y, ddof=1)
            ax.errorbar(mx, my, xerr=sx, yerr=sy, fmt="o", markersize=7,
                        mfc=col, mec="black", ecolor=col, elinewidth=1.2, capsize=3)
        if n >= 3:
            confidence_ellipse(x, y, ax, level=0.95, edgecolor=col)
```

```
    if title:
        ax.set_title(title, fontsize=STYLE["legend_title_fontsize"])


# Figure
def plot_isotopes_white_vs_brown_mixed(df_in, titles=None, xylabels=None,
                                       custom_palette=None, legend_labels=None,
                                       xlim=(-34, -18), ylim=(-10, 8)):
    palette = grouped_palette(custom_palette)
    if legend_labels is None:
        legend_labels = {}

    default_titles = {
        (0, 0): r"$\it{Trametes}$ - hardwood",
        (0, 1): r"$\it{Fomes\ fomentarius}$ - hardwood",
        (1, 0): r"$\it{Fomitopsis\ pinicola}$ - softwood",
        (1, 1): r"$\it{Fomitopsis\ pinicola}$ - hardwood",
    }
    if titles:
        default_titles.update(titles)

    labels = {"x": "δ13C [‰]", "y": "δ15N [‰]"}
    if xylabels:
        labels.update(xylabels)

    fig, axes = plt.subplots(2, 2, figsize=(13, 11), dpi=400, sharex=True, sharey=True)

    panel_specs = {
        (0, 0): dict(species=["trametes"], substrate="hardwood", split_fruiting=False),
        (0, 1): dict(species=["fomes fomentarius", "fomes"], substrate="hardwood", split_fruiting=False),
        (1, 0): dict(species=["fomitopsis pinicola", "fomitopsis"], substrate="softwood",
split_fruiting=True),
        (1, 1): dict(species=["fomitopsis pinicola", "fomitopsis"], substrate="hardwood",
split_fruiting=True),
    }

    # plot each panel
    for (r, c), spec in panel_specs.items():
        plot_isotopes_on_ax(df_in, ax=axes[r, c], palette=palette,
                            title=default_titles[(r, c)], **spec)

    axes[0, 0].set_xlim(*xlim)
    axes[0, 0].set_ylim(*ylim)

    # axis labeling
    for c in range(2):
        axes[1, c].set_xlabel(labels["x"], fontsize=STYLE["xlabel_fontsize"], color=STYLE["label_color"])
        axes[0, c].tick_params(labelbottom=False)
    for r in range(2):
        axes[r, 0].set_ylabel(labels["y"], fontsize=STYLE["ylabel_fontsize"], color=STYLE["label_color"])
        axes[r, 1].tick_params(labelleft=False)

    # === Legends ===
    white = ["fruiting body", "wood with mycelia", "wood without mycelia",
             "weathered wood", "bark"]
    handles_white = [
        Line2D([], [], color=palette[c], marker=COMPARTMENT_MARKERS[c],
               linestyle="", markersize=7, label=legend_labels.get(c, c))
        for c in white if c in palette
    ]
    axes[0, 1].legend(handles=handles_white, loc="upper right",
                      fontsize=STYLE["legend_fontsize"], frameon=True)

    brown = ["pileipellis", "trama", "hymenium",
             "wood with mycelia", "wood without mycelia", "weathered wood", "bark"]
    handles_brown = [
        Line2D([], [], color=palette[c], marker=COMPARTMENT_MARKERS[c],
               linestyle="", markersize=7, label=legend_labels.get(c, c))
        for c in brown if c in palette
    ]
    axes[1, 0].legend(handles=handles_brown, loc="lower left",
                      fontsize=STYLE["legend_fontsize"], frameon=True)

    plt.tight_layout()
    plt.show()


# printing results
titles_iso = {
    (0, 0): r"$\it{Trametes}$ growing on hardwood",
    (0, 1): r"$\it{Fomes\ fomentarius}$ growing on hardwood",
    (1, 0): r"$\it{Fomitopsis\ pinicola}$ growing on softwood",
    (1, 1): r"$\it{Fomitopsis\ pinicola}$ growing on hardwood",
}

xylabels_iso = {"x": "δ13C [‰]", "y": "δ15N [‰]"}

legend_labels = {
```

```
    "pileipellis": "pileipellis",
    "trama": "trama",
    "hymenium": "hymenium",
    "fruiting body": "basidiocarp",
    "wood with mycelia": "wood with mycelia",
    "wood without mycelia": "wood without mycelia",
    "weathered wood": "weathered wood",
    "bark": "bark",
}

plot_isotopes_white_vs_brown_mixed(
    df2,
    titles=titles_iso,
    xylabels=xylabels_iso,
    custom_palette=None,
    legend_labels=legend_labels,
)
```

#DRIFT Analysis: Mean +- SD values and figure visualization

```
import pandas as pd
import numpy as np
import re
import matplotlib.pyplot as plt
from pathlib import Path
from scipy.signal import savgol_filter
import unicodedata

# file path to fetch the data
FILE = r"250807_Drift_Data_RP.xlsx"
SHEET = "Baseline_Corrected"
df = pd.read_excel(FILE, sheet_name=SHEET, index_col=0)
print(df.head())

# figure style
STYLE = {
    "font_family": "sans-serif", "font_sans": ["Arial"],"xlabel_fontsize": 18,"ylabel_fontsize":
18,"xtick_fontsize": 15,"ytick_fontsize": 15,"legend_fontsize": 15,"legend_title_fontsize":
16,"legend_frameon": True,"label_color": "black",
}

# group assignment
raw = df.copy()
wn = pd.to_numeric(raw.index.to_series(), errors="coerce").to_numpy()
spec = raw.apply(pd.to_numeric, errors="coerce")

mask = ~np.all(np.isnan(spec.to_numpy()), axis=1)
wn, spec = wn[mask], spec.loc[mask, :]
order = np.argsort(wn)
wn, spec = wn[order], spec.iloc[order, :]

sample_names = list(spec.columns.astype(str))

def extract_sample_num(name):
    m = re.search(r"(\d+)$", str(name))
    return int(m.group(1)) if m else None

sample_nums = [extract_sample_num(s) for s in sample_names]

fruiting = [1, 2, 23, 36, 44, 75, 86, 93, 100, 107, 24, 35, 45, 76, 87, 94, 101, 108,
            25, 34, 43, 77, 88, 95, 102, 109, 3, 19, 61, 4, 18, 62, 33, 66, 67, 9, 11, 32, 64, 63]
wood_with = [7, 17, 58, 22, 39, 42, 91, 80, 85, 98, 105, 51, 69, 73, 28, 30, 46]
wood_without = [5, 15, 59, 21, 37, 92, 81, 84, 99, 106, 52, 70, 74, 26, 29, 47, 53, 55]
bark = [8, 13, 57, 14, 40, 89, 78, 82, 96, 103, 49, 71, 10, 12, 65]
weathered_wood = [6, 16, 20, 27, 31, 38, 41, 48, 50, 54, 56, 60, 68, 72, 79, 83, 90, 97, 104]

id_to_group = {}
for sid in fruiting:        id_to_group[sid] = "basidiocarp"
for sid in wood_with:       id_to_group[sid] = "wood with mycelia"
for sid in wood_without:    id_to_group[sid] = "wood without mycelia"
for sid in bark:            id_to_group[sid] = "bark"
for sid in weathered_wood:  id_to_group[sid] = "weathered wood"

groups = [id_to_group.get(n, None) for n in sample_nums]
print("\nGroup counts:")
print(pd.Series(groups, dtype="object").value_counts(dropna=False))

# spectra
X = wn.astype(float)
Y = spec.to_numpy().astype(float)

smooth = True
if smooth and len(X) > 21:
    for j in range(Y.shape[1]):
        y = Y[:, j]
        ok = np.isfinite(y)
        if ok.sum() > 21:
```

```
            Y[ok, j] = savgol_filter(y[ok], window_length=21, polyorder=3)

# band definitions

BANDS_ALL = {
    "Aliphatic C-H":         (3040, 2760),
    "Carboxylic C=O":        (1765, 1700),
    "Aromatic C=C (2)":      (1690, 1600),
    "Lignin-like residues": (1550, 1485),
    "Aromatic C=C (1)":      (1395, 1350),
    "Polysaccharide":        (1080, 1020),
    "Aromatic C-H":          (830, 761),
}

def _band_mask(lo, hi, X):
    lo_, hi_ = (lo, hi) if lo <= hi else (hi, lo)
    return (X >= lo_) & (X <= hi_)

# mean ± SD per band and group
def compute_band_group_stats(X, Y, groups, bands=BANDS_ALL, ddof=0):
    X, Y = np.asarray(X), np.asarray(Y)
    groups = np.asarray(groups, dtype=object)
    valid = np.array([g is not None and g != "" for g in groups])
    Y, groups = Y[:, valid], groups[valid]
    rows = []
    uniq = pd.unique(groups)
    for band, (hi, lo) in bands.items():
        m = _band_mask(lo, hi, X)
        if not np.any(m):
            continue
        band_vals = np.nanmean(Y[m, :], axis=0)
        for g in uniq:
            idx = np.where(groups == g)[0]
            if idx.size == 0:
                continue
            vals = band_vals[idx]
            rows.append({
                "band": band,
                "group": str(g),
                "n": int(idx.size),
                "mean": float(np.nanmean(vals)),
                "sd": float(np.nanstd(vals, ddof=ddof)),
            })
    return (pd.DataFrame(rows)
            .sort_values(["band", "group"])
            .reset_index(drop=True))

stats_long = compute_band_group_stats(X, Y, groups, BANDS_ALL)
stats_long["formatted"] = (
    stats_long["mean"].round(4).astype(str)
    + " ± " + stats_long["sd"].round(4).astype(str)
    + " (n=" + stats_long["n"].astype(str) + ")"
)
print("\n=== Mean ± SD per Band and Compartment ===")
print(stats_long[["band", "group", "formatted"]])
stats_long.to_excel("drift_mean_sd_by_compartment.xlsx", index=False)


# plotting results
color_map = {
    "basidiocarp": "#1f4e79",
    "wood with mycelia": "#6baed6",
    "wood without mycelia": "#e6550d",
    "weathered wood": "#fd8d3c",
    "bark": "#9467bd",
}

def add_bands_minimal(ax, bands, face="#b0b0b0", edge="#8c8c8c",
                      alpha=0.10, edge_lw=0.6, fs=12, y_frac=0.5):
    y0, y1 = ax.get_ylim()
    y = y0 + y_frac * (y1 - y0)
    for label, (hi, lo) in bands.items():
        lo_, hi_ = min(lo, hi), max(lo, hi)
        ax.axvspan(lo_, hi_, color=face, alpha=alpha, lw=0, zorder=0)
        ax.axvline(lo_, color=edge, lw=edge_lw, alpha=0.8, zorder=1)
        ax.axvline(hi_, color=edge, lw=edge_lw, alpha=0.8, zorder=1)
        mid = 0.5 * (lo_ + hi_)
        ax.text(mid, y, label, ha="center", va="center",
                fontsize=fs, color="black", rotation=90,
                bbox=dict(fc="white", ec="none", alpha=0.25))

def plot_mean_sd(ax, X, Y, groups, color_map, lw=2.0, sd_alpha=0.15):
    for g, col in color_map.items():
        idx = [i for i, gg in enumerate(groups) if gg == g]
        if not idx:
            continue
        m = np.nanmean(Y[:, idx], axis=1)
        s = np.nanstd(Y[:, idx], axis=1)
```

```
        ax.plot(X, m, color=col, lw=lw, label=g)
        ax.fill_between(X, m - s, m + s, color=col, alpha=sd_alpha, linewidth=0)

def split_bands_for_window(bands, wmin=1000, wmax=2000):
    lo, hi = min(wmin, wmax), max(wmin, wmax)
    outside, inside = {}, {}
    for name, (b_hi, b_lo) in bands.items():
        b_lo_, b_hi_ = min(b_lo, b_hi), max(b_lo, b_hi)
        overlaps = (b_lo_ <= hi) and (b_hi_ >= lo)
        if overlaps:
            inside[name] = (b_hi, b_lo)
        else:
            outside[name] = (b_hi, b_lo)
    return outside, inside

bands_outside, _ = split_bands_for_window(BANDS_ALL, 1000, 2000)
_, bands_zoom_fp = split_bands_for_window(BANDS_ALL, 1000, 1800)

fig, axs = plt.subplots(2, 1, figsize=(12, 12))

# --- Panel (a): overview ---
plot_mean_sd(axs[0], X, Y, groups, color_map)
axs[0].invert_xaxis()
add_bands_minimal(axs[0], bands_outside, fs=12, y_frac=0.35)
axs[0].set_xlabel("wavenumber (cm$^{-1}$)", fontsize=STYLE["xlabel_fontsize"])
axs[0].set_ylabel("absorbance", fontsize=STYLE["ylabel_fontsize"])
axs[0].tick_params(axis="both", labelsize=STYLE["xtick_fontsize"])
axs[0].set_title("(a) Overview", fontsize=STYLE["legend_title_fontsize"])
axs[0].legend(
    title="Compartment",
    loc="upper left",
    frameon=STYLE["legend_frameon"],
    fontsize=STYLE["legend_fontsize"],
    title_fontsize=STYLE["legend_title_fontsize"],
)

# --- Panel (b): fingerprint region ---
plot_mean_sd(axs[1], X, Y, groups, color_map)
axs[1].set_xlim(1000, 1800)
axs[1].invert_xaxis()
add_bands_minimal(axs[1], bands_zoom_fp, fs=12, y_frac=0.16)
axs[1].set_xlabel("wavenumber (cm$^{-1}$)", fontsize=STYLE["xlabel_fontsize"])
axs[1].tick_params(axis="both", labelsize=STYLE["xtick_fontsize"])
axs[1].set_title("(b) Fingerprint region", fontsize=STYLE["legend_title_fontsize"])

plt.tight_layout()
plt.savefig("drift_spectra_overview_fingerprint.png", dpi=400, bbox_inches="tight")
plt.show()
```

 # DRIFT Analysis: Mean ± SD values of organic compound groups and PCA tables

```
import re
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from matplotlib.lines import Line2D
from matplotlib.patches import Patch
import matplotlib.patheffects as pe
from adjustText import adjust_text

# figure style
STYLE = {
    "font_family": "sans-serif","font_sans": ["Arial"],"xlabel_fontsize": 14,"ylabel_fontsize":
14,"xtick_fontsize": 14,"ytick_fontsize": 14,"legend_fontsize": 14,"legend_title_fontsize":
16,"legend_frameon": True,"label_color": "black",
}

# file path to fetch the data
FILE = r"250807_Drift_Data_RP.xlsx"
SHEET = "AUC"

df = pd.read_excel(FILE, sheet_name=SHEET, index_col=0)
df = df.T.reset_index().rename(columns={"index": "Sample"})

def extract_sample_num(name):
    m = re.search(r"(\d+)$", str(name))
    return int(m.group(1)) if m else None

df["Sample_Num"] = df["Sample"].apply(extract_sample_num)

# group assignment
white_rot_fungi = [3, 19, 61, 4, 18, 62, 33, 66, 67, 9, 11, 32]
white_rot_wood_with = [28, 30, 46, 51, 69, 73, 7, 17, 58]
white_rot_wood_without = [26, 29, 47, 52, 70, 74, 5, 15, 59]
white_rot_bark = [10, 12, 49, 71, 8, 13, 57]
```

```
white_rot_weathered_wood = [27, 31, 48, 50, 68, 72, 6, 16, 60]
brown_rot_fungi = [
    23, 36, 44, 93, 24, 35, 45, 94, 25, 34, 43, 95,
    75, 86, 100, 107, 76, 87, 101, 108, 77, 88, 102, 109]
brown_rot_wood_with = [80, 85, 98, 105, 22, 39, 42, 91]
brown_rot_wood_without = [21, 37, 92, 81, 84, 99, 106]
brown_rot_bark = [14, 40, 89, 78, 82, 96, 103]
brown_rot_weathered_wood = [79, 83, 97, 104, 20, 38, 41, 90]

def assign_group(n):
    # White-rot
    if n in white_rot_fungi:
        return "White-rot fruiting body"
    if n in white_rot_wood_with:
        return "White-rot wood with mycelia"
    if n in white_rot_wood_without:
        return "White-rot wood without mycelia"
    if n in white_rot_weathered_wood:
        return "White-rot weathered wood"
    if n in white_rot_bark:
        return "White-rot bark"

    # Brown-rot
    if n in brown_rot_fungi:
        return "Brown-rot fruiting body"
    if n in brown_rot_wood_with:
        return "Brown-rot wood with mycelia"
    if n in brown_rot_wood_without:
        return "Brown-rot wood without mycelia"
    if n in brown_rot_weathered_wood:
        return "Brown-rot weathered wood"
    if n in brown_rot_bark:
        return "Brown-rot bark"

    return None

df["Group"] = df["Sample_Num"].apply(assign_group)

def split_group(g):
    if g is None:
        return (None, None)
    rot = "White-rot" if g.startswith("White-rot") else ("Brown-rot" if g.startswith("Brown-rot") else
None)
    comp = g.replace("White-rot ", "").replace("Brown-rot ", "")
    return rot, comp

df[["RotType", "Compartment_raw"]] = df["Group"].apply(split_group).apply(pd.Series)

# Merge all fruiting-body tissues to Basidiocarp
df["Compartment"] = df["Compartment_raw"].replace({"fruiting body": "Basidiocarp"})

print("\nGroup counts:")
print(df["Group"].value_counts(dropna=False))

# Species assignment for basidiocarps
trametes_basi = [66, 67, 9, 11, 32, 33]
fomes_basi = [3, 19, 61, 4, 18, 62]

def assign_species(n, rottype, compartment):
    if compartment != "Basidiocarp":
        return None
    if rottype == "Brown-rot" and n in brown_rot_fungi:
        return "Fomitopsis pinicola"
    if rottype == "White-rot":
        if n in trametes_basi:
            return "Trametes spp."
        if n in fomes_basi:
            return "Fomes fomentarius"
    return None

df["Species"] = [
    assign_species(n, r, c)
    for n, r, c in zip(df["Sample_Num"], df["RotType"], df["Compartment"])
]

drop_cols = [
    c for c in [
        "Sample", "Sample_Num", "Group", "RotType",
        "Compartment_raw", "Compartment", "Species", "AUC_sum"
    ]
    if c in df.columns
]

feat_df = df.drop(columns=drop_cols)
feat_df = feat_df.apply(pd.to_numeric, errors="coerce")
feat_df = feat_df.loc[:, feat_df.notna().any(axis=0)]
feature_names = feat_df.columns.to_numpy()
```

```
# Keep only rows with group assignment
mask_valid = df["Group"].notna()
df = df.loc[mask_valid].copy()
feat_df = feat_df.loc[mask_valid].copy()

brown_rot_triplets = [
    (23, 24, 25),
    (36, 35, 34),
    (44, 45, 43),
    (93, 94, 95),
    (75, 76, 77),
    (86, 87, 88),
    (100, 101, 102),
    (107, 108, 109),
]

bio_map = {}
for k, (a, b, c) in enumerate(brown_rot_triplets, start=1):
    for n in (a, b, c):
        bio_map[n] = f"BR{k:02d}"

df["BioUnit"] = df["Sample_Num"].map(bio_map).fillna(df["Sample_Num"].astype(str))

is_basi = (df["Compartment"] == "Basidiocarp") & df["RotType"].notna()

collapsed_basi = (
    df.loc[is_basi, ["RotType", "Species", "Compartment", "BioUnit"]]
      .join(feat_df.loc[is_basi])
      .groupby(["RotType", "Species", "Compartment", "BioUnit"], as_index=False)
      .mean(numeric_only=True)
)

kept_nonbasi = (
    df.loc[~is_basi, ["RotType", "Species", "Compartment", "BioUnit"]]
      .join(feat_df.loc[~is_basi])
      .reset_index(drop=True)
)

combined = pd.concat([collapsed_basi, kept_nonbasi], ignore_index=True)

# coloring and markers
color_map_rot = {
    "White-rot": "#1f77b4",
    "Brown-rot": "#8c564b",
}
color_map_species = {
    "Fomitopsis pinicola": "#8c564b",
    "Trametes spp.": "#87CEFA",
    "Fomes fomentarius": "#00008B",
}
marker_map_comp = {
    "Basidiocarp": "o",
    "wood with mycelia": "D",
    "wood without mycelia": "P",
    "weathered wood": "^",
    "bark": "s",
}
label_map_exact = {
    "Aromatic_CH_range2 / 806 - 830 cm-1": "Arom2 C-H",
    "Aromatic_CH_range1 / 761 - 806 cm-1": "Arom1 C-H",
    "Aromatic_CC_range_2 / 1600 - 1690 cm-1": "Arom2 C=C",
    "Aromatic_CC_range_1 / 1350 - 1395 cm-1": "Arom1 C=C",
    "Carboxylic_range / 1700 - 1765 cm-1": "Carboxylic",
    "Aliphatic_range / 2760 - 3040 cm-1": "Aliphatic",
    "Lignin_range / 1485 - 1550 cm-1": "Lignin",
    "Phenolic_cellulose_range / NULL": "Cellulose",
    "Polysaccharide_range / 1020 - 1080 cm-1": "Polysaccharide",
    "Arom C-H": "Arom C-H",
}

def pretty_label(name):
    return label_map_exact.get(str(name), str(name))

def italic_species(name):
    name = name.replace(" ", r"\ ")
    return rf"$\it{{{name}}}$"

# column names
col_arom_ch_1 = "Aromatic_CH_range1 / 761 - 806 cm-1"
col_arom_ch_2 = "Aromatic_CH_range2 / 806 - 830 cm-1"

col_arom_cc_1 = "Aromatic_CC_range_1 / 1350 - 1395 cm-1"
col_arom_cc_2 = "Aromatic_CC_range_2 / 1600 - 1690 cm-1"
col_carbox = "Carboxylic_range / 1700 - 1765 cm-1"
col_aliph = "Aliphatic_range / 2760 - 3040 cm-1"
col_poly = "Polysaccharide_range / 1020 - 1080 cm-1"
col_cellulose = "Phenolic_cellulose_range / NULL"
```

```
# Panel B: only keep aromatic, carboxylic, aliphatic and cellulose
panelB_keep_cols = [
    col_arom_cc_1,
    col_arom_cc_2,
    col_carbox,
    col_aliph,
    col_cellulose
]

# Matrix builders
def make_panelA_matrix(df_plot):
    meta_cols = ["RotType", "Species", "Compartment", "BioUnit"]
    Xdf = df_plot.drop(columns=meta_cols, errors="ignore").copy()

    have1 = col_arom_ch_1 in Xdf.columns
    have2 = col_arom_ch_2 in Xdf.columns

    if have1 and have2:
        Xdf["Arom C-H"] = Xdf[col_arom_ch_1].fillna(0) + Xdf[col_arom_ch_2].fillna(0)
        Xdf = Xdf.drop(columns=[col_arom_ch_1, col_arom_ch_2])
    elif have1:
        Xdf = Xdf.rename(columns={col_arom_ch_1: "Arom C-H"})
    elif have2:
        Xdf = Xdf.rename(columns={col_arom_ch_2: "Arom C-H"})

    feat_names_A = Xdf.columns.to_numpy()
    X = Xdf.to_numpy()
    return Xdf, X, feat_names_A

def make_panelB_matrix(df_plot):
    available = [c for c in panelB_keep_cols if c in df_plot.columns]
    Xdf = df_plot[available].copy()

    rename_map = {
        col_arom_cc_1: "Arom1 C=C",
        col_arom_cc_2: "Arom2 C=C",
        col_carbox: "Carboxylic",
        col_aliph: "Aliphatic",
        col_cellulose: "Cellulose",
    }
    Xdf = Xdf.rename(columns=rename_map)

    feat_names_B = Xdf.columns.to_numpy()
    X = Xdf.to_numpy()
    return Xdf, X, feat_names_B

# PCA helpers
def pca_scores_and_loadings(X):
    Xs = StandardScaler().fit_transform(X)
    pca = PCA()
    scores = pca.fit_transform(Xs)
    load = pca.components_.T * np.sqrt(pca.explained_variance_)
    mags = np.linalg.norm(load[:, :2], axis=1)
    return pca, scores, load, mags

def pca_appendix_tables(Xdf, meta_df, panel_name, out_prefix):
    """
    Create appendix-ready PCA tables:
    1) explained variance / eigenvalues
    2) loadings
    3) scores
    """
    Xs = StandardScaler().fit_transform(Xdf.to_numpy())
    pca = PCA()
    scores = pca.fit_transform(Xs)

    # ---- Table 1: explained variance ----
    explained_df = pd.DataFrame({
        "PC": [f"PC{i+1}" for i in range(len(pca.explained_variance_))],
        "Standard deviation": np.sqrt(pca.explained_variance_),
        "Eigenvalue": pca.explained_variance_,
        "Proportion of variance (%)": pca.explained_variance_ratio_ * 100,
        "Cumulative variance (%)": np.cumsum(pca.explained_variance_ratio_) * 100
    })

    # ---- Table 2: loadings (= eigenvectors / variable weights) ----
    loadings_df = pd.DataFrame(
        pca.components_.T,
        index=Xdf.columns,
        columns=[f"PC{i+1}" for i in range(pca.components_.shape[0])]
    ).reset_index().rename(columns={"index": "Variable"})

    # ---- Table 3: scores ----
    scores_df = pd.concat(
        [
            meta_df.reset_index(drop=True),
            pd.DataFrame(
                scores,
```

```
                columns=[f"PC{i+1}" for i in range(scores.shape[1])]
            )
        ],
        axis=1
    )

    explained_file = f"{out_prefix}_explained_variance.xlsx"
    loadings_file = f"{out_prefix}_loadings.xlsx"
    scores_file = f"{out_prefix}_scores.xlsx"

    explained_df.to_excel(explained_file, index=False)
    loadings_df.to_excel(loadings_file, index=False)
    scores_df.to_excel(scores_file, index=False)

    print(f"\n=== {panel_name}: PCA explained variance ===")
    print(explained_df)

    print(f"\n=== {panel_name}: PCA loadings (first columns) ===")
    print(loadings_df)

    print(f"\nSaved PCA appendix tables for {panel_name}:")
    print(explained_file)
    print(loadings_file)
    print(scores_file)

    return explained_df, loadings_df, scores_df

# Plot helpers
def draw_biplot_panel_A(ax, df_plot, title):
    Xdf, X, feature_names_A = make_panelA_matrix(df_plot)
    rot = df_plot["RotType"].to_numpy()
    comp = df_plot["Compartment"].to_numpy()

    pca, scores, load, mags = pca_scores_and_loadings(X)

    for r in ["White-rot", "Brown-rot"]:
        for c in marker_map_comp.keys():
            sel = (rot == r) & (comp == c)
            if np.any(sel):
                ax.scatter(
                    scores[sel, 0], scores[sel, 1],
                    c=color_map_rot[r],
                    marker=marker_map_comp[c],
                    s=115,
                    edgecolor="black",
                    linewidths=0.5,
                    alpha=0.9,
                    zorder=3
                )

    max_loading = np.max(np.linalg.norm(load[:, :2], axis=1))
    max_score = np.max(np.abs(scores[:, :2]))
    scale = 0.35 * (max_score / max_loading) if max_loading > 0 else 1.0

    texts = []
    for i in range(load.shape[0]):
        x, y = load[i, 0] * scale, load[i, 1] * scale

        ann = ax.annotate(
            "", xy=(x, y), xytext=(0, 0),
            arrowprops=dict(arrowstyle="-|>", lw=2.6, color="black", alpha=0.75),
            zorder=2
        )
        if ann.arrow_patch is not None:
            ann.arrow_patch.set_path_effects([
                pe.Stroke(linewidth=3.2, foreground="white"),
                pe.Normal()
            ])

        t = ax.text(
            x * 1.0, y * 1.1,
            pretty_label(feature_names_A[i]),
            fontsize=12,
            ha="center",
            va="center",
            zorder=5,
            clip_on=False
        )
        t.set_path_effects([
            pe.Stroke(linewidth=3, foreground="white"),
            pe.Normal()
        ])
        texts.append(t)

    ax.figure.canvas.draw()
    adjust_text(
        texts,
        ax=ax,
```

```
            expand_text=(1.2, 1.2),
            expand_points=(1.2, 1.2),
            force_text=0.4,
            force_points=0.2,
            only_move={"text": "xy"},
            arrowprops=None
        )

        ax.axhline(0, color="0.85", lw=1.2, zorder=1)
        ax.axvline(0, color="0.85", lw=1.2, zorder=1)
        ax.set_xlabel(
            f"PC1 ({pca.explained_variance_ratio_[0] * 100:.1f}%)",
            fontsize=STYLE["xlabel_fontsize"],
            color=STYLE["label_color"]
        )
        ax.set_ylabel(
            f"PC2 ({pca.explained_variance_ratio_[1] * 100:.1f}%)",
            fontsize=STYLE["ylabel_fontsize"],
            color=STYLE["label_color"]
        )
        ax.set_title(title, fontsize=STYLE["legend_title_fontsize"])
        ax.tick_params(axis="x", labelsize=STYLE["xtick_fontsize"])
        ax.tick_params(axis="y", labelsize=STYLE["ytick_fontsize"])

    def draw_biplot_panel_B(ax, df_plot, title):
        Xdf, X, feature_names_B = make_panelB_matrix(df_plot)
        species = df_plot["Species"].to_numpy()

        pca, scores, load, mags = pca_scores_and_loadings(X)

        for sp in [s for s in color_map_species.keys() if s in set(species)]:
            sel = (species == sp)
            if np.any(sel):
                ax.scatter(
                    scores[sel, 0], scores[sel, 1],
                    c=color_map_species[sp],
                    marker="o",
                    s=115,
                    edgecolor="black",
                    linewidths=0.5,
                    alpha=0.9,
                    zorder=3
                )

        max_loading = np.max(np.linalg.norm(load[:, :2], axis=1))
        max_score = np.max(np.abs(scores[:, :2]))
        base_scale = (max_score / max_loading) if max_loading > 0 else 1.0

        scale = 0.65 * base_scale
        label_r = 1.0

        texts = []
        for i in range(load.shape[0]):
            x, y = load[i, 0] * scale, load[i, 1] * scale

            ann = ax.annotate(
                "", xy=(x, y), xytext=(0, 0),
                arrowprops=dict(arrowstyle="-|>", lw=2.6, color="black", alpha=0.75),
                zorder=2
            )
            if ann.arrow_patch is not None:
                ann.arrow_patch.set_path_effects([
                    pe.Stroke(linewidth=3.2, foreground="white"),
                    pe.Normal()
                ])

            t = ax.text(
                x * label_r, y * label_r,
                str(feature_names_B[i]),
                fontsize=12,
                ha="center",
                va="center",
                zorder=5,
                clip_on=False
            )
            t.set_path_effects([
                pe.Stroke(linewidth=3, foreground="white"),
                pe.Normal()
            ])
            texts.append(t)

        ax.figure.canvas.draw()
        adjust_text(
            texts,
            ax=ax,
            expand_text=(2.0, 2.0),
            expand_points=(2.0, 2.0),
            force_text=1.2,
```

```
        force_points=0.6,
        only_move={"text": "xy"},
        arrowprops=None
    )

    ax.axhline(0, color="0.85", lw=1.2, zorder=1)
    ax.axvline(0, color="0.85", lw=1.2, zorder=1)
    ax.set_xlabel(
        f"PC1 ({pca.explained_variance_ratio_[0] * 100:.1f}%)",
        fontsize=STYLE["xlabel_fontsize"],
        color=STYLE["label_color"]
    )
    ax.set_ylabel(
        f"PC2 ({pca.explained_variance_ratio_[1] * 100:.1f}%)",
        fontsize=STYLE["ylabel_fontsize"],
        color=STYLE["label_color"]
    )
    ax.set_title(title, fontsize=STYLE["legend_title_fontsize"])
    ax.tick_params(axis="x", labelsize=STYLE["xtick_fontsize"])
    ax.tick_params(axis="y", labelsize=STYLE["ytick_fontsize"])

# Building data for panels
df_A = combined.copy()
df_B = collapsed_basi.copy()
df_B = df_B[df_B["Compartment"] == "Basidiocarp"].copy()

# mean ± SD tables
def compute_mean_sd_table(df_in, group_cols, value_cols=None, ddof=0):
    d = df_in.copy()

    if value_cols is None:
        value_cols = d.select_dtypes(include=[np.number]).columns.tolist()

    value_cols = [c for c in value_cols if c not in group_cols]

    rows = []
    grouped = d.groupby(group_cols, dropna=False)

    for keys, subdf in grouped:
        if not isinstance(keys, tuple):
            keys = (keys,)

        group_info = dict(zip(group_cols, keys))

        for col in value_cols:
            vals = pd.to_numeric(subdf[col], errors="coerce").dropna()
            if len(vals) == 0:
                mean_val = np.nan
                sd_val = np.nan
                n_val = 0
            else:
                mean_val = float(np.mean(vals))
                sd_val = float(np.std(vals, ddof=ddof))
                n_val = int(len(vals))

            rows.append({
                **group_info,
                "variable": col,
                "n": n_val,
                "mean": mean_val,
                "sd": sd_val,
                "formatted": (
                    f"{mean_val:.4f} ± {sd_val:.4f} (n={n_val})"
                    if n_val > 0 else "NA"
                )
            })

    return pd.DataFrame(rows)

# --- Table A: all compartments ---
meta_cols_A = ["RotType", "Species", "Compartment", "BioUnit"]
value_cols_A = [c for c in df_A.columns if c not in meta_cols_A]

stats_A = compute_mean_sd_table(
    df_A,
    group_cols=["RotType", "Compartment"],
    value_cols=value_cols_A,
    ddof=0
)

print("\n=== Mean ± SD for all compartments (grouped by RotType and Compartment) ===")
print(stats_A[["RotType", "Compartment", "variable", "formatted"]])

stats_A.to_excel("PCA_mean_sd_all_compartments.xlsx", index=False)

# --- Table B: basidiocarp only ---
meta_cols_B = ["RotType", "Species", "Compartment", "BioUnit"]
value_cols_B = [c for c in df_B.columns if c not in meta_cols_B]
```

```
stats_B = compute_mean_sd_table(
    df_B,
    group_cols=["Species"],
    value_cols=value_cols_B,
    ddof=0
)

print("\n=== Mean ± SD for basidiocarp only (grouped by Species) ===")
print(stats_B[["Species", "variable", "formatted"]])

stats_B.to_excel("PCA_mean_sd_basidiocarp_by_species.xlsx", index=False)

# PCA tables (explained variance, cumulative variance)
Xdf_A, X_A, feat_names_A = make_panelA_matrix(df_A)
meta_A = df_A[["RotType", "Species", "Compartment", "BioUnit"]].copy()

explained_A, loadings_A, scores_A = pca_appendix_tables(
    Xdf_A,
    meta_A,
    panel_name="Panel A (all compartments)",
    out_prefix="PCA_table_A"
)

Xdf_B, X_B, feat_names_B = make_panelB_matrix(df_B)
meta_B = df_B[["RotType", "Species", "Compartment", "BioUnit"]].copy()

explained_B, loadings_B, scores_B = pca_appendix_tables(
    Xdf_B,
    meta_B,
    panel_name="Panel B (basidiocarp only)",
    out_prefix="PCA_table_B"
)

# Plotting results
fig, axes = plt.subplots(2, 1, figsize=(8, 12), dpi=400)

draw_biplot_panel_A(
    axes[0],
    df_A,
    "A) All compartments"
)

draw_biplot_panel_B(
    axes[1],
    df_B,
    "B) Basidiocarp only"
)

# --- Legends ---
handles_left = [
    Line2D([], [], color="black", marker=m,
           linestyle="None", markersize=10, label=l)
    for l, m in marker_map_comp.items()
]

handles_mid = [
    Patch(color=color_map_rot["White-rot"], label="White-rot"),
    Patch(color=color_map_rot["Brown-rot"], label="Brown-rot"),
]

present_species = [s for s in color_map_species.keys() if s in set(df_B["Species"].dropna())]
handles_right = [
    Patch(color=color_map_species[s], label=italic_species(s))
    for s in present_species
]

fig.legend(
    handles=handles_left,
    title="Compartment",
    loc="upper left",
    bbox_to_anchor=(0.97, 0.87),
    frameon=STYLE["legend_frameon"],
    fontsize=STYLE["legend_fontsize"],
    title_fontsize=STYLE["legend_title_fontsize"],
    ncol=1
)

fig.legend(
    handles=handles_mid,
    title="Rot type",
    loc="upper center",
    bbox_to_anchor=(0.5, 0.92),
    frameon=False,
    fontsize=STYLE["legend_fontsize"],
    title_fontsize=STYLE["legend_title_fontsize"],
    ncol=2
)
```

```
fig.legend(
    handles=handles_right,
    title="Basidiocarp",
    loc="upper right",
    bbox_to_anchor=(1.33, 0.41),
    frameon=STYLE["legend_frameon"],
    fontsize=STYLE["legend_fontsize"],
    title_fontsize=STYLE["legend_title_fontsize"],
    ncol=1
)

plt.tight_layout(rect=[0, 0, 1, 0.88])
plt.show()

print("\nCollapsed basidiocarps (BioUnits) per species:")
print(df_B["Species"].value_counts(dropna=False))
```

#Fatty acid analysis: visualization of FA ratios

```
#import packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl

# file path to fetch the data
FA_FILE = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Ladina\new\Calculations_FA_Ladina_Reich.xlsx"
FA_SHEET = "FA_results"

fa = pd.read_excel(FA_FILE, sheet_name=FA_SHEET)

COMP_COL = "Compartment"

fa[COMP_COL] = fa[COMP_COL].astype(str).str.strip().str.lower()

# figure style
mpl.rcParams["font.family"] = "sans-serif"
mpl.rcParams["font.sans-serif"] = ["Arial"]

plt.rcParams.update({
    "font.size": 14,
    "axes.titlesize": 16,
    "axes.labelsize": 15,
    "xtick.labelsize": 13,
    "ytick.labelsize": 13,
})

comp_colors = {
    "pileipellis": "#6b79d6",
    "trama": "#377eb8",
    "hymenium": "#69c3a3",
    "wood with mycelia": "#6baed6",
    "wood without mycelia": "#e6550d",
    "weathered wood": "#fd8d3c",
    "bark": "#9467bd",
}

# mapping basidiocarp compartments
comp_map = {
    "fruiting body outside": "pileipellis",
    "fruiting body inside": "trama",
    "fruiting body spores": "hymenium",
}

fa[COMP_COL] = fa[COMP_COL].replace(comp_map)

existing = [
    "pileipellis",
    "trama",
    "hymenium",
    "wood with mycelia",
    "wood without mycelia",
    "weathered wood",
    "bark",
]

fa[COMP_COL] = pd.Categorical(fa[COMP_COL], categories=existing, ordered=True)

# Defining free extractable fatty acid groups
BCFA = ["BCFA_17", "BCFA_23", "BCFA_25"]
DCA  = ["DCA_9", "DCA_16", "DCA_20", "DCA_22"]

SSCFA = ["C14:0", "C15:0", "C16:0", "C17:0", "C18:0", "C20:0"]
SLCFA = ["C21:0", "C22:0", "C23:0", "C24:0", "C25:0", "C26:0", "C28:0"]
```

```
USCFA = ["C15:1", "C16:1", "C16:2", "C18:1", "C18:2"]
ULCFA = ["C24:1", "C25:1", "C26:1"]

GROUPS = {
    "BCFA": BCFA,
    "DCA": DCA,
    "SSCFA": SSCFA,
    "SLCFA": SLCFA,
    "USCFA": USCFA,
    "ULCFA": ULCFA,
}


# Checking columns and compute group sums
for gname, cols in GROUPS.items():
    valid_cols = [c for c in cols if c in fa.columns]
    missing_cols = [c for c in cols if c not in fa.columns]

    if missing_cols:
        print(f"Warning for {gname}: missing columns -> {missing_cols}")

    if len(valid_cols) == 0:
        fa[gname] = np.nan
    else:
        fa[gname] = fa[valid_cols].sum(axis=1)

# Computing ratios
eps = 1e-12

fa["SC_FA"] = fa["SSCFA"] + fa["USCFA"]
fa["LC_FA"] = fa["SLCFA"] + fa["ULCFA"]

fa["SAT_FA"]   = fa["SSCFA"] + fa["SLCFA"]
fa["UNSAT_FA"] = fa["USCFA"] + fa["ULCFA"]

fa["ratio_LC_SC"]     = fa["LC_FA"] / (fa["SC_FA"] + eps)
fa["ratio_UNSAT_SAT"] = fa["UNSAT_FA"] / (fa["SAT_FA"] + eps)

# only keep rows valid for ratio calculation
fa_ratio = fa[(fa["SC_FA"] > 0) & (fa["SAT_FA"] > 0)].copy()


# Create mean ± SD summary table by compartment
summary_vars = [
    "BCFA", "DCA", "SSCFA", "SLCFA", "USCFA", "ULCFA",
    "ratio_LC_SC", "ratio_UNSAT_SAT"
]

summary_stats = (
    fa.groupby(COMP_COL, observed=True)[summary_vars]
      .agg(["mean", "std"])
      .reindex(existing)
)

summary_table = pd.DataFrame(index=summary_vars, columns=existing)

for var in summary_vars:
    for comp in existing:
        try:
            mean_val = summary_stats.loc[comp, (var, "mean")]
            sd_val   = summary_stats.loc[comp, (var, "std")]

            if pd.isna(mean_val):
                summary_table.loc[var, comp] = ""
            elif pd.isna(sd_val):
                summary_table.loc[var, comp] = f"{mean_val:.3f} ± 0.000"
            else:
                summary_table.loc[var, comp] = f"{mean_val:.3f} ± {sd_val:.3f}"
        except KeyError:
            summary_table.loc[var, comp] = ""

print("\nMean ± SD table:")
print(summary_table)


# Compute y-limits
def compute_ylim(df, col):
    vals = df[col].dropna().values
    if len(vals) == 0:
        return (0, 1)

    pad = 0.1 * (vals.max() - vals.min()) if vals.max() > vals.min() else 0.2
    return vals.min() - pad, vals.max() + pad

ymin_lcsc, ymax_lcsc = compute_ylim(fa_ratio, "ratio_LC_SC")
ymin_us, ymax_us = compute_ylim(fa_ratio, "ratio_UNSAT_SAT")
```

```
# Plot function with colored boxplots
def plot_ratio_boxpanel(ax, df, col, ylabel, ylim):
    data = []
    labels = []
    colors = []

    for comp in existing:
        vals = df.loc[df[COMP_COL] == comp, col].dropna().values
        if len(vals) == 0:
            continue
        data.append(vals)
        labels.append(comp)
        colors.append(comp_colors.get(comp, "#bdbdbd"))

    bp = ax.boxplot(
        data,
        patch_artist=True,
        widths=0.65,
        showfliers=True,
        medianprops=dict(color="black", linewidth=1.4),
        whiskerprops=dict(color="black", linewidth=1.0),
        capprops=dict(color="black", linewidth=1.0),
        boxprops=dict(edgecolor="black", linewidth=1.0),
        flierprops=dict(
            marker="o",
            markersize=4,
            markerfacecolor="white",
            markeredgecolor="black",
            alpha=0.8
        )
    )

    for patch, c in zip(bp["boxes"], colors):
        patch.set_facecolor(c)
        patch.set_alpha(0.85)

    ax.set_xticks(np.arange(1, len(labels) + 1))
    ax.set_xticklabels(labels, rotation=45, ha="right")
    ax.set_ylabel(ylabel)
    ax.set_ylim(ylim)
    ax.grid(False)

# Create figure
fig, axes = plt.subplots(2, 1, figsize=(12, 8), sharex=True)

plot_ratio_boxpanel(
    axes[0],
    fa_ratio,
    "ratio_UNSAT_SAT",
    "Unsaturated / saturated FA",
    (ymin_us, ymax_us)
)

plot_ratio_boxpanel(
    axes[1],
    fa_ratio,
    "ratio_LC_SC",
    "Long-chain / short-chain FA",
    (ymin_lcsc, ymax_lcsc)
)

plt.tight_layout()

plt.savefig(
    "free_extractable_FA_ratios_by_compartment_boxplot.png",
    dpi=400,
    bbox_inches="tight"
)

plt.show()
```

#Fatty acid analysis: visualization of the FA C16:1, C18:1, C18:2, C26:1

```
# import packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

# file path to fetch the data
FA_FILE = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Ladina\new\Calculations_FA_Ladina_Reich.xlsx"
FA_SHEET = "FA_results"

# column names
COMP_COL = "Compartment"
```

```
SUBSTRATE_COL = "Substrate_type"
TREE_COL = "Tree_ID"

FA_COLS = ["C16:1", "C18:1", "C18:2", "C26:1"]

fa = pd.read_excel(FA_FILE, sheet_name=FA_SHEET)

df = fa.copy()

# figure style
plt.rcParams.update({
    "font.size": 10,
    "axes.titlesize": 11,
    "axes.labelsize": 10,
})

sub_colors = {"hardwood": "#c8c08b", "softwood": "#9c6427"}
sub_labels = {"hardwood": "Hardwood", "softwood": "Softwood"}

# cleaning columns
df[COMP_COL] = df[COMP_COL].astype(str).str.strip()

rename_map = {
    "fruiting body outside": "pileipellis",
    "fruiting body inside": "trama",
    "fruiting body spores": "hymenium",
}
df[COMP_COL] = df[COMP_COL].replace(rename_map)

df[SUBSTRATE_COL] = (
    df[SUBSTRATE_COL]
    .astype(str).str.strip().str.lower()
    .replace({
        "hard wood": "hardwood",
        "soft wood": "softwood"
    })
)

df = df[df[SUBSTRATE_COL].isin(["hardwood", "softwood"])].copy()
df[TREE_COL] = df[TREE_COL].astype(str).str.strip()

for col in FA_COLS:
    df[col] = pd.to_numeric(df[col], errors="coerce")

df = df.dropna(subset=[COMP_COL, SUBSTRATE_COL, TREE_COL]).copy()

compartment_order = [
    "pileipellis", "trama", "hymenium",
    "wood with mycelia", "wood without mycelia",
    "weathered wood", "bark",
]
compartment_order = [c for c in compartment_order if c in df[COMP_COL].unique()]

df[COMP_COL] = pd.Categorical(df[COMP_COL], categories=compartment_order, ordered=True)

substrate_order = ["hardwood", "softwood"]
df[SUBSTRATE_COL] = pd.Categorical(df[SUBSTRATE_COL], categories=substrate_order, ordered=True)

# compute mean +- SD
def compute_stats_table(df, value_col):
    tmp = df.dropna(subset=[value_col]).copy()
    return (
        tmp.groupby([SUBSTRATE_COL, COMP_COL], observed=True)[value_col]
           .agg(["mean", "std"])
           .reset_index()
    )

stats_tables = {fa_col: compute_stats_table(df, fa_col) for fa_col in FA_COLS}

formatted_rows = []

for fa_col in FA_COLS:
    for sub in substrate_order:
        row = {"Fatty acid": f"{fa_col} ({sub})"}
        sub_stats = stats_tables[fa_col]

        for comp in compartment_order:
            tmp = sub_stats[
                (sub_stats[SUBSTRATE_COL] == sub) &
                (sub_stats[COMP_COL] == comp)
            ]

            if tmp.empty:
                row[comp] = ""
            else:
                mean_val = tmp["mean"].iloc[0]
                sd_val = tmp["std"].iloc[0]
                if pd.isna(sd_val):
```

```
                    sd_val = 0
                row[comp] = f"{mean_val:.3f} ± {sd_val:.3f}"

        formatted_rows.append(row)

summary_fmt = pd.DataFrame(formatted_rows)

print("\n=== Final mean ± SD table ===")
print(summary_fmt)

# plot panel
def plot_fa_panel(ax, stats_tbl, title, show_xticklabels=True):
    x = np.arange(len(compartment_order))
    bar_width = 0.38
    offsets = {"hardwood": -bar_width/2, "softwood": bar_width/2}

    ymax = 0

    for sub in substrate_order:
        sub_stats = stats_tbl[stats_tbl[SUBSTRATE_COL] == sub]
        sub_stats = sub_stats.set_index(COMP_COL).reindex(compartment_order).reset_index()

        means = sub_stats["mean"].fillna(0).to_numpy()
        sds = sub_stats["std"].fillna(0).to_numpy()
        xpos = x + offsets[sub]

        ax.bar(xpos, means, width=bar_width,
               color=sub_colors[sub], edgecolor="black", linewidth=0.4)

        ax.errorbar(xpos, means, yerr=sds, fmt="none",
                    ecolor="black", elinewidth=1, capsize=3)

        ymax = max(ymax, np.max(means + sds))

    ax.set_title(title)
    ax.set_xticks(x)

    if show_xticklabels:
        ax.set_xticklabels(compartment_order, rotation=45, ha="right")
    else:
        ax.set_xticklabels([])

    # y scaling
    ax.set_ylim(0, ymax * 1.1 if ymax > 0 else 1)
    ax.set_yticks(np.linspace(0, ax.get_ylim()[1], 5))
    ax.yaxis.set_major_formatter(
        FuncFormatter(lambda x, pos: f"{x:.3f}".rstrip("0").rstrip("."))
    )

# printing results
fig, axes = plt.subplots(2, 2, figsize=(9.45, 5.75), dpi=400)
axes = axes.flatten()

plot_fa_panel(axes[0], stats_tables["C16:1"], "C16:1", False)
plot_fa_panel(axes[1], stats_tables["C18:1"], "C18:1", False)
plot_fa_panel(axes[2], stats_tables["C18:2"], "C18:2", True)
plot_fa_panel(axes[3], stats_tables["C26:1"], "C26:1", True)

fig.supylabel("Concentration [nmol g$^{-1}$ dry mass]", x=0.02)

handles = [
    plt.Rectangle((0,0),1,1,color=sub_colors["hardwood"], ec="black"),
    plt.Rectangle((0,0),1,1,color=sub_colors["softwood"], ec="black")
]

fig.legend(handles, ["Hardwood", "Softwood"],
           title="Substrate",
           loc="upper center", ncol=2, frameon=False)

plt.subplots_adjust(left=0.08, bottom=0.20, top=0.86)

plt.savefig("FA_plot.png", dpi=400, bbox_inches="tight")
plt.show()
```

#Sterol analysis: mean +- SD and figure visualization

```
#importing packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# file path to fetch the data
STEROL_FILE = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Calculations_sterols_Ladina_Reich.xlsx"
STEROL_SHEET = "sterol_results"
```

```
# figure style
plt.rcParams.update({
    "font.size": 14,"axes.titlesize": 14,"axes.labelsize": 13,"xtick.labelsize": 12,"ytick.labelsize":
12,"legend.fontsize": 12,"legend.title_fontsize": 13,
})

sub_colors = {
    "hardwood": "#C9C28E",
    "softwood": "#8B5A2B",
}

sub_labels = {
    "hardwood": "Hardwood",
    "softwood": "Softwood",
}

# column names
COMP_COL = "Compartment"
SUBSTRATE_COL = "Substrate_type"
TREE_COL = "Tree_ID"
ERG_COL = "Ergosterol"

OTHER_STEROL_COLS = [
    "Erg.2 (Dehydroergosterol)",
    "β-Sitosterol",
    "Stigmastanol",
    "γ-Sitostenone",
]

# reading data
sterol = pd.read_excel(STEROL_FILE, sheet_name=STEROL_SHEET)

need_cols = [COMP_COL, SUBSTRATE_COL, TREE_COL, ERG_COL]
missing = [c for c in need_cols if c not in sterol.columns]
if missing:
    raise KeyError(f"Missing columns in Excel: {missing}")

missing_other = [c for c in OTHER_STEROL_COLS if c not in sterol.columns]
if missing_other:
    print(f"Warning: these sterol columns were not found and will be skipped: {missing_other}")

present_other_sterols = [c for c in OTHER_STEROL_COLS if c in sterol.columns]

df = sterol.copy()

# cleaning columns
df[COMP_COL] = df[COMP_COL].astype(str).str.strip()

rename_map = {
    "fruiting body outside": "pileipellis",
    "fruiting body inside": "trama",
    "fruiting body spores": "hymenium",
}
df[COMP_COL] = df[COMP_COL].replace(rename_map)

df[SUBSTRATE_COL] = (
    df[SUBSTRATE_COL]
    .astype(str).str.strip().str.lower()
    .replace({
        "hard wood": "hardwood",
        "soft wood": "softwood",
        "hardwood": "hardwood",
        "softwood": "softwood"
    })
)

df = df[df[SUBSTRATE_COL].isin(["hardwood", "softwood"])].copy()

df[TREE_COL] = df[TREE_COL].astype(str).str.strip()

df[ERG_COL] = pd.to_numeric(df[ERG_COL], errors="coerce")
for col in present_other_sterols:
    df[col] = pd.to_numeric(df[col], errors="coerce")

df = df.dropna(subset=[COMP_COL, SUBSTRATE_COL, TREE_COL]).copy()

compartment_order = [
    "pileipellis",
    "trama",
    "hymenium",
    "wood with mycelia",
    "wood without mycelia",
    "weathered wood",
    "bark",
]
compartment_order = [c for c in compartment_order if c in df[COMP_COL].unique()]

df[COMP_COL] = pd.Categorical(
```

```
    df[COMP_COL],
    categories=compartment_order,
    ordered=True
)

substrate_order = ["hardwood", "softwood"]
df[SUBSTRATE_COL] = pd.Categorical(
    df[SUBSTRATE_COL],
    categories=substrate_order,
    ordered=True
)

# tables: Ergosterol: separated by substrate and compartment
erg_df = df.dropna(subset=[ERG_COL]).copy()

erg_stats = (
    erg_df.groupby([SUBSTRATE_COL, COMP_COL])[ERG_COL]
          .agg(["mean", "std", "count"])
          .reset_index()
)

print("\n=== Ergosterol mean ± SD by substrate and compartment ===")
print(erg_stats)

# formatted Ergosterol table
erg_table = pd.DataFrame(index=["Ergosterol (hardwood)", "Ergosterol (softwood)"],
                         columns=compartment_order)

for sub in substrate_order:
    row_name = f"Ergosterol ({sub})"
    for comp in compartment_order:
        tmp = erg_stats[
            (erg_stats[SUBSTRATE_COL] == sub) &
            (erg_stats[COMP_COL] == comp)
        ]
        if tmp.empty:
            erg_table.loc[row_name, comp] = ""
        else:
            mean_val = tmp["mean"].iloc[0]
            sd_val = tmp["std"].iloc[0]
            if pd.isna(sd_val):
                sd_val = 0
            erg_table.loc[row_name, comp] = f"{mean_val:.3f} ± {sd_val:.3f}"


# table: other sterols: by compartment only
other_tables = []

for sterol_col in present_other_sterols:
    tmp_df = df.dropna(subset=[sterol_col]).copy()

    tmp_stats = (
        tmp_df.groupby(COMP_COL)[sterol_col]
              .agg(["mean", "std", "count"])
              .reindex(compartment_order)
    )

    row = {}
    for comp in compartment_order:
        if comp not in tmp_stats.index or pd.isna(tmp_stats.loc[comp, "mean"]):
            row[comp] = ""
        else:
            mean_val = tmp_stats.loc[comp, "mean"]
            sd_val = tmp_stats.loc[comp, "std"]
            if pd.isna(sd_val):
                sd_val = 0
            row[comp] = f"{mean_val:.3f} ± {sd_val:.3f}"

    other_tables.append(pd.Series(row, name=sterol_col))

other_table = pd.DataFrame(other_tables)

# combined formatted table
final_table = pd.concat([erg_table, other_table], axis=0)

print("\n=== Final sterol summary table (mean ± SD) ===")
print(final_table)

# printing Barplot panel
fig, ax = plt.subplots(figsize=(10, 6), dpi=400)

x = np.arange(len(compartment_order))
bar_width = 0.38

offsets = {
    "hardwood": -bar_width / 2,
    "softwood": +bar_width / 2
}
```

```
for sub in substrate_order:
    sub_stats = erg_stats[erg_stats[SUBSTRATE_COL] == sub].copy()
    sub_stats[COMP_COL] = pd.Categorical(
        sub_stats[COMP_COL],
        categories=compartment_order,
        ordered=True
    )
    sub_stats = sub_stats.sort_values(COMP_COL)

    sub_stats = (
        sub_stats.set_index(COMP_COL)
                 .reindex(compartment_order)
                 .reset_index()
    )

    means = sub_stats["mean"].fillna(0).to_numpy()
    sds = sub_stats["std"].fillna(0).to_numpy()

    xpos = x + offsets[sub]

    ax.bar(
        xpos,
        means,
        width=bar_width,
        color=sub_colors[sub],
        edgecolor="black",
        linewidth=0.4,
        label=sub_labels[sub],
        zorder=2
    )

    ax.errorbar(
        xpos,
        means,
        yerr=sds,
        fmt="none",
        ecolor="black",
        elinewidth=1.1,
        capsize=3,
        capthick=1.1,
        zorder=3
    )

ax.set_xticks(x)
ax.set_xticklabels(compartment_order, rotation=45, ha="right")
ax.set_ylabel("Ergosterol concentration [nmol g$^{-1}$ dry mass]")

handles, labels = ax.get_legend_handles_labels()
fig.legend(
    handles, labels,
    title="Substrate type",
    loc="upper center",
    ncol=2,
    frameon=False
)

ax.grid(False)

plt.tight_layout(rect=[0, 0, 1, 0.93])
out_png = "Ergosterol_by_compartment_substrate_barplot.png"
fig.savefig(out_png, dpi=400, bbox_inches="tight")
plt.show()
print(f"\nSaved plot: {out_png}")
```

 # Correlation analysis: Spearman correlation between log10(Ergosterol) and log10 fatty acids (C18:2, C18:1, C26:1, C16:1)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import spearmanr, norm

# file paths to fetch the data
STEROL_FILE = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Calculations_sterols_Ladina_Reich.xlsx"
STEROL_SHEET = "sterol_results"
FA_FILE = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Ladina\new\Calculations_FA_Ladina_Reich.xlsx"
FA_SHEET = "FA_results"

# figure style
plt.rcParams.update({
    "font.size": 10,
    "axes.titlesize": 11,
    "axes.labelsize": 10,
    "xtick.labelsize": 10,
```

```
    "ytick.labelsize": 10,
    "legend.fontsize": 10,
    "legend.title_fontsize": 10,
})
sub_colors = {
    "hardwood": "#c8c08b",
    "softwood": "#9c6427"
}
sub_labels = {
    "hardwood": "Hardwood",
    "softwood": "Softwood"
}

# Column names
ID_COL = "Sample_ID"
COMP_COL = "Compartment"
SUBSTRATE_COL = "Substrate_type"
TREE_COL = "Tree_ID"
ERG_COL = "Ergosterol"
FA_COLS = ["C18:2", "C18:1", "C26:1", "C16:1"]

# Read data
sterol = pd.read_excel(STEROL_FILE, sheet_name=STEROL_SHEET)
fa = pd.read_excel(FA_FILE, sheet_name=FA_SHEET)

# Clean sterol data
sterol_df = sterol.copy()

sterol_df[ID_COL] = sterol_df[ID_COL].astype(str).str.strip()
sterol_df[COMP_COL] = sterol_df[COMP_COL].astype(str).str.strip()

rename_map = {
    "fruiting body outside": "pileipellis",
    "fruiting body inside": "trama",
    "fruiting body spores": "hymenium",
}
sterol_df[COMP_COL] = sterol_df[COMP_COL].replace(rename_map)

sterol_df[SUBSTRATE_COL] = (
    sterol_df[SUBSTRATE_COL]
    .astype(str).str.strip().str.lower()
    .replace({
        "hard wood": "hardwood",
        "soft wood": "softwood",
        "hardwood": "hardwood",
        "softwood": "softwood"
    })
)

sterol_df[TREE_COL] = sterol_df[TREE_COL].astype(str).str.strip()
sterol_df[ERG_COL] = pd.to_numeric(sterol_df[ERG_COL], errors="coerce")

sterol_df = sterol_df.dropna(subset=[ID_COL, COMP_COL, SUBSTRATE_COL, TREE_COL]).copy()
sterol_df = sterol_df[sterol_df[SUBSTRATE_COL].isin(["hardwood", "softwood"])].copy()

# Clean FA data
fa_df = fa.copy()

fa_df[ID_COL] = fa_df[ID_COL].astype(str).str.strip()
fa_df[COMP_COL] = fa_df[COMP_COL].astype(str).str.strip()
fa_df[COMP_COL] = fa_df[COMP_COL].replace(rename_map)

fa_df[SUBSTRATE_COL] = (
    fa_df[SUBSTRATE_COL]
    .astype(str).str.strip().str.lower()
    .replace({
        "hard wood": "hardwood",
        "soft wood": "softwood",
        "hardwood": "hardwood",
        "softwood": "softwood"
    })
)

fa_df[TREE_COL] = fa_df[TREE_COL].astype(str).str.strip()

for col in FA_COLS:
    fa_df[col] = pd.to_numeric(fa_df[col], errors="coerce")

fa_df = fa_df.dropna(subset=[ID_COL, COMP_COL, SUBSTRATE_COL, TREE_COL]).copy()
fa_df = fa_df[fa_df[SUBSTRATE_COL].isin(["hardwood", "softwood"])].copy()

# merge data
sterol_keep = [ID_COL, COMP_COL, SUBSTRATE_COL, TREE_COL, ERG_COL]
fa_keep = [ID_COL] + FA_COLS

data = pd.merge(
    sterol_df[sterol_keep],
    fa_df[fa_keep],
```

```
    on=ID_COL,
    how="inner"
)

print("\nMerged rows:", len(data))
print("Compartments included after merge:")
print(sorted(data[COMP_COL].dropna().unique()))

# Helper
def add_regression_line(ax, x, y, color="black", lw=1.3):
    """Add simple linear regression line to existing axis."""
    if len(x) >= 2:
        coeffs = np.polyfit(x, y, 1)
        xx = np.linspace(np.nanmin(x), np.nanmax(x), 100)
        yy = coeffs[0] * xx + coeffs[1]
        ax.plot(xx, yy, color=color, linewidth=lw, zorder=2)

def fisher_z_from_r(r):
    """Convert correlation coefficient to Fisher z."""
    return 0.5 * np.log((1 + r) / (1 - r))

# Plotting results
results = []

fig, axes = plt.subplots(2, 2, figsize=(9.45, 7.2), dpi=400)
axes = axes.flatten()

for ax, fa_col in zip(axes, FA_COLS):

    # panel-specific subset:
    # keep all rows with ergosterol + THIS fatty acid
    panel_df = data[[COMP_COL, SUBSTRATE_COL, ERG_COL, fa_col]].copy()
    panel_df = panel_df.dropna(subset=[ERG_COL, fa_col, SUBSTRATE_COL])

    # only positive values for log10
    panel_df = panel_df[(panel_df[ERG_COL] > 0) & (panel_df[fa_col] > 0)].copy()

    # log-transform
    panel_df["log10_Erg"] = np.log10(panel_df[ERG_COL])
    panel_df["log10_FA"] = np.log10(panel_df[fa_col])

    # --------------------------------
    # overall Spearman correlation
    # --------------------------------
    n_total = len(panel_df)

    if n_total >= 2:
        rho_total, p_total = spearmanr(panel_df["log10_Erg"], panel_df["log10_FA"])
    else:
        rho_total, p_total = np.nan, np.nan

    # --------------------------------
    # substrate-specific correlations
    # --------------------------------
    sub_stats = {}

    for sub in ["hardwood", "softwood"]:
        tmp = panel_df[panel_df[SUBSTRATE_COL] == sub].copy()
        n_sub = len(tmp)

        if n_sub >= 2:
            rho_sub, p_sub = spearmanr(tmp["log10_Erg"], tmp["log10_FA"])
        else:
            rho_sub, p_sub = np.nan, np.nan

        sub_stats[sub] = {
            "n": n_sub,
            "rho": rho_sub,
            "p": p_sub
        }

    # --------------------------------
    # Fisher z-test: hardwood vs softwood
    # --------------------------------
    rho_hw = sub_stats["hardwood"]["rho"]
    rho_sw = sub_stats["softwood"]["rho"]
    n_hw = sub_stats["hardwood"]["n"]
    n_sw = sub_stats["softwood"]["n"]

    if (
        np.isfinite(rho_hw) and np.isfinite(rho_sw)
        and abs(rho_hw) < 1 and abs(rho_sw) < 1
        and n_hw > 3 and n_sw > 3
    ):
        z_hw = fisher_z_from_r(rho_hw)
        z_sw = fisher_z_from_r(rho_sw)

        fisher_z = (z_hw - z_sw) / np.sqrt(1 / (n_hw - 3) + 1 / (n_sw - 3))
```

```
            p_fisher = 2 * (1 - norm.cdf(abs(fisher_z)))
        else:
            fisher_z = np.nan
            p_fisher = np.nan

        # --------------------------------
        # store results for table
        # --------------------------------
        results.append({
            "Fatty acid": fa_col,
            "rho_total": rho_total,
            "p_total": p_total,
            "n_total": n_total,
            "rho_hardwood": sub_stats["hardwood"]["rho"],
            "p_hardwood": sub_stats["hardwood"]["p"],
            "n_hardwood": sub_stats["hardwood"]["n"],
            "rho_softwood": sub_stats["softwood"]["rho"],
            "p_softwood": sub_stats["softwood"]["p"],
            "n_softwood": sub_stats["softwood"]["n"],
            "fisher_z": fisher_z,
            "p_fisher": p_fisher,
        })

        # --------------------------------
        # plot scatter by substrate
        # --------------------------------
        for sub in ["hardwood", "softwood"]:
            tmp = panel_df[panel_df[SUBSTRATE_COL] == sub].copy()

            ax.scatter(
                tmp["log10_Erg"],
                tmp["log10_FA"],
                s=28,
                color=sub_colors[sub],
                edgecolor="black",
                linewidth=0.3,
                alpha=0.9
            )

        # overall regression line
        add_regression_line(
            ax,
            panel_df["log10_Erg"].to_numpy(),
            panel_df["log10_FA"].to_numpy(),
            color="black",
            lw=1.3
        )

        # panel annotation: total n + total rho only
        rho_text = f"{rho_total:.2f}" if np.isfinite(rho_total) else "NA"
        ax.text(
            0.05, 0.95,
            f"n = {n_total}\nρ = {rho_text}",
            transform=ax.transAxes,
            ha="left",
            va="top",
            fontsize=10
        )

        ax.set_title(fa_col)
        ax.set_xlabel("log10(Ergosterol)")
        ax.set_ylabel(f"log10({fa_col})")
        ax.grid(False)

    # legend
    handles = [
        plt.Line2D([0], [0], marker='o', color='w',
                   markerfacecolor=sub_colors["hardwood"],
                   markeredgecolor="black", markeredgewidth=0.4,
                   markersize=6, label="Hardwood"),
        plt.Line2D([0], [0], marker='o', color='w',
                   markerfacecolor=sub_colors["softwood"],
                   markeredgecolor="black", markeredgewidth=0.4,
                   markersize=6, label="Softwood"),
    ]

    fig.legend(
        handles, ["Hardwood", "Softwood"],
        title="Substrate type",
        loc="upper center",
        ncol=2,
        frameon=False
    )

    plt.tight_layout(rect=[0, 0, 1, 0.93])

    out_png = "Ergosterol_FA_correlation_panel_all_compartments.png"
    fig.savefig(out_png, dpi=400, bbox_inches="tight")
```

```
plt.show()

print(f"\nSaved plot: {out_png}")

# Create and save results table
results_df = pd.DataFrame(results)
results_df_rounded = results_df.copy()
for col in results_df_rounded.columns:
    if col != "Fatty acid":
        results_df_rounded[col] = pd.to_numeric(results_df_rounded[col], errors="coerce").round(3)

print("\n=== Correlation + Fisher z-test results ===")
print(results_df_rounded.to_string(index=False))

# save table
out_xlsx = "Ergosterol_FA_correlation_stats.xlsx"
results_df_rounded.to_excel(out_xlsx, index=False)

print(f"\nSaved table: {out_xlsx}")
```

EA, DRIFT and Lipid biomarker results in a PCA combined

```
# importing packages: numpy, pandas, matplotlib
import re
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.lines as mlines
import matplotlib.patches as mpatches
import matplotlib.patheffects as pe
from adjustText import adjust_text

# file paths to fetch the data
ORGANIC_FILE  = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\332B25LR_DRIFT\250807_Drift_Data_Ladina_Reich.xlsx"
ORGANIC_SHEET = "AUC"
FA_FILE  = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Ladina\new\Calculations_FA_Ladina_Reich.xlsx"
FA_SHEET = "FA_results"
STEROL_FILE  = r"C:\Users\Ladina\OneDrive - Universität Zürich
UZH\Master\Masterarbeit\Calculations_sterols_Ladina_Reich.xlsx"
STEROL_SHEET = "sterol_results"

# Figure style
color_map = {"hardwood": "#C9C28E", "softwood": "#8B5A2B"}

marker_left = {
    "Basidiocarp": "o",
    "Wood with mycelia": "D",
    "Wood without mycelia": "P",
    "Weathered wood": "^",
    "Bark": "s",
}
marker_right = {"Pileipellis": "o", "Trama": "^", "Hymenium": "s"}

plt.rcParams.update({
    "font.size": 12,
    "axes.titlesize": 14,
    "axes.labelsize": 14,
    "xtick.labelsize": 14,
    "ytick.labelsize": 14,
    "legend.fontsize": 11,
    "legend.title_fontsize": 12,
})

label_map_exact = {
    "Aromatic_CH_range2 / 806 - 830 cm-1": "Arom2 C-H",
    "Aromatic_CH_range1 / 761 - 806 cm-1": "Arom1 C-H",
    "Aromatic_CC_range_2 / 1600 - 1690 cm-1": "Arom2 C=C",
    "Aromatic_CC_range_1 / 1350 - 1395 cm-1": "Arom1 C=C",
    "Carboxylic_range / 1700 - 1765 cm-1": "Carboxylic",
    "Aliphatic_range / 2760 - 3040 cm-1": "Aliphatic",
    "Lignin_range / 1485 - 1550 cm-1": "Lignin",
    "Phenolic_cellulose_range / NULL": "Cellulose",
    "Polysaccharide_range / 1020 - 1080 cm-1": "Polysaccharide",
    "Arom C-H": "Arom C-H",
    "Erg.2 (Dehydroergosterol)": "Erg.2"
}

def pretty_label(x):
    return label_map_exact.get(str(x), str(x))

# setting
DO_MEAN_IMPUTE = True
```

```
PRINT_DEBUG = True
TOPK_LABELS = None
LABEL_FONTSIZE = 12
ARROW_LW = 1.6
ARROW_ALPHA = 0.80
ARROW_HEAD = 11
LABEL_OFFSET = 1.10
LABEL_STROKE = 2.6
ARROW_STROKE = 2.4
LIMIT_PAD = 0.08

# Panel-specific removal lists
PANEL_A_DROP_VARS = {"tree_id", "Tree_ID", "AUC_sum", "auc_sum"}
PANEL_B_DROP_FA_GROUPS = {"FA_BCFA", "FA_DCA"}
PANEL_B_DROP_STEROLS = {"Stigmastanol", "β-Sitosterol", "γ-Sitostenone"}
PANEL_B_DROP_ORGANICS = {
    "Lignin",
    "Cellulose",
    "Lignin_range / 1485 - 1550 cm-1",
    "Phenolic_cellulose_range / NULL",
}

# helpers
def normalize_id(x):
    if pd.isna(x):
        return np.nan
    return re.sub(r"\s+", "", str(x).strip().lower())

def normalize_text(x):
    if pd.isna(x):
        return np.nan
    s = str(x).strip().lower()
    s = s.replace("-", "-").replace("—", "-").replace("_", "-")
    s = re.sub(r"\s+", " ", s)
    return s

def harmonize_substrate(x):
    if pd.isna(x):
        return "unknown"
    s = normalize_text(x)
    if s in {"hw", "h w"}:
        return "hardwood"
    if s in {"sw", "s w"}:
        return "softwood"
    if "hardwood" in s or "hard wood" in s:
        return "hardwood"
    if "softwood" in s or "soft wood" in s:
        return "softwood"
    if any(k in s for k in ["broadleaf", "deciduous", "angiosperm"]):
        return "hardwood"
    if any(k in s for k in ["conifer", "coniferous", "gymnosperm"]):
        return "softwood"
    return "unknown"

def harmonize_compartment(x):
    s = normalize_text(x)
    if pd.isna(s):
        return np.nan
    rename_map = {
        "fruiting body outside": "pileipellis",
        "fruiting body inside": "trama",
        "fruiting body spores": "hymenium",
    }
    s = rename_map.get(s, s)
    if s.startswith("fruiting body"):
        if "outside" in s:
            return "pileipellis"
        if "inside" in s:
            return "trama"
        if "spores" in s:
            return "hymenium"
        return "basidiocarp"
    if s in ["sporocarp", "basidiocarp"]:
        return "basidiocarp"
    return s

def to_num(x):
    if pd.isna(x):
        return np.nan
    s = str(x).strip().replace(",", ".")
    s = "".join(ch for ch in s if ch.isdigit() or ch in ".-+eE")
    return pd.to_numeric(s, errors="coerce")

def mean_impute_df(Xdf):
    X = Xdf.copy()
    for c in X.columns:
        if X[c].isna().any():
            X[c] = X[c].fillna(X[c].mean(skipna=True))
```

```
    return X

def pca_scores_and_loadings(X):
    Xs = StandardScaler().fit_transform(X)
    pca = PCA(n_components=2, random_state=0)
    scores = pca.fit_transform(Xs)
    load = pca.components_.T * np.sqrt(pca.explained_variance_)
    return pca, scores, load

def label_var(v):
    v = str(v)
    if v.startswith("FA_"):
        v = v.replace("FA_", "")
    return pretty_label(v)

def coarse_compartment(c):
    c = harmonize_compartment(c)
    if c in ["pileipellis", "trama", "hymenium", "basidiocarp"]:
        return "Basidiocarp"
    if c == "wood with mycelia":
        return "Wood with mycelia"
    if c == "wood without mycelia":
        return "Wood without mycelia"
    if c == "weathered wood":
        return "Weathered wood"
    if c == "bark":
        return "Bark"
    return None

def prepare_matrix(df, feature_cols, panel_name=""):
    X = df[feature_cols].apply(pd.to_numeric, errors="coerce")
    X = X.replace([np.inf, -np.inf], np.nan)

    all_nan_cols = X.columns[X.isna().all()].tolist()
    if all_nan_cols and PRINT_DEBUG:
        print(f"[{panel_name}] Dropping all-NaN columns:", all_nan_cols)
    X = X.drop(columns=all_nan_cols)

    if DO_MEAN_IMPUTE:
        X = mean_impute_df(X)

    return X.dropna(axis=0, how="any")

def expand_limits_to_texts(ax, texts, pad=0.08):
    if not texts:
        return

    fig = ax.figure
    fig.canvas.draw()
    renderer = fig.canvas.get_renderer()

    xmins, xmaxs, ymins, ymaxs = [], [], [], []
    for t in texts:
        if not t.get_visible():
            continue
        bb = t.get_window_extent(renderer=renderer)
        (x0, y0) = ax.transData.inverted().transform((bb.x0, bb.y0))
        (x1, y1) = ax.transData.inverted().transform((bb.x1, bb.y1))
        xmins.append(min(x0, x1))
        xmaxs.append(max(x0, x1))
        ymins.append(min(y0, y1))
        ymaxs.append(max(y0, y1))

    if not xmins:
        return

    xmin, xmax = min(xmins), max(xmaxs)
    ymin, ymax = min(ymins), max(ymaxs)

    cur_xmin, cur_xmax = ax.get_xlim()
    cur_ymin, cur_ymax = ax.get_ylim()

    xmin = min(xmin, cur_xmin)
    xmax = max(xmax, cur_xmax)
    ymin = min(ymin, cur_ymin)
    ymax = max(ymax, cur_ymax)

    xspan = xmax - xmin
    yspan = ymax - ymin
    ax.set_xlim(xmin - pad * xspan, xmax + pad * xspan)
    ax.set_ylim(ymin - pad * yspan, ymax + pad * yspan)

# load organic compounds
org = pd.read_excel(ORGANIC_FILE, sheet_name=ORGANIC_SHEET)

required_org = ["Sample_ID", "Species_type", "Substrate_type", "Compartment"]
missing_req = [c for c in required_org if c not in org.columns]
if missing_req:
```

```
    raise KeyError(f"Organic AUC sheet is missing columns: {missing_req}")

org["Sample_ID"] = org["Sample_ID"].apply(normalize_id)
org["Substrate_type"] = org["Substrate_type"].apply(harmonize_substrate)
org["Compartment"] = org["Compartment"].apply(harmonize_compartment)

for c in ["Tree_ID", "tree_id", "AUC_sum", "auc_sum"]:
    if c in org.columns:
        org = org.drop(columns=[c])

org_meta_cols = ["Sample_ID", "Species_type", "Substrate_type", "Compartment"]
org_feat_cols = [c for c in org.columns if c not in org_meta_cols]

col_arom_ch_1 = "Aromatic_CH_range1 / 761 - 806 cm-1"
col_arom_ch_2 = "Aromatic_CH_range2 / 806 - 830 cm-1"
if col_arom_ch_1 in org.columns and col_arom_ch_2 in org.columns:
    org["Arom C-H"] = org[col_arom_ch_1].fillna(0) + org[col_arom_ch_2].fillna(0)
    org = org.drop(columns=[col_arom_ch_1, col_arom_ch_2])
    org_feat_cols = [c for c in org_feat_cols if c not in [col_arom_ch_1, col_arom_ch_2]] + ["Arom C-H"]

for c in org_feat_cols:
    org[c] = org[c].apply(to_num)

org_df = org[org_meta_cols + org_feat_cols].drop_duplicates(subset=["Sample_ID"]).set_index("Sample_ID")

# load FA
fa = pd.read_excel(FA_FILE, sheet_name=FA_SHEET)
if "Sample_ID" not in fa.columns:
    raise KeyError("FA sheet must contain column 'Sample_ID'.")

fa["Sample_ID"] = fa["Sample_ID"].apply(normalize_id)

BCFA = ["BCFA_17", "BCFA_23", "BCFA_25"]
DCA = ["DCA_9", "DCA_16", "DCA_20", "DCA_22"]
SSCFA = ["C14:0", "C15:0", "C16:0", "C17:0", "C18:0"]
SLCFA = ["C20:0", "C21:0", "C22:0", "C23:0", "C24:0", "C25:0", "C26:0", "C28:0"]
USCFA = ["C15:1", "C16:1", "C16:2", "C18:1", "C18:2"]
ULCFA = ["C24:1", "C25:1", "C26:1"]

FA_GROUPS = {
    "FA_BCFA": BCFA,
    "FA_DCA": DCA,
    "FA_SSCFA": SSCFA,
    "FA_SLCFA": SLCFA,
    "FA_USCFA": USCFA,
    "FA_ULCFA": ULCFA,
}

for cols in FA_GROUPS.values():
    for c in cols:
        if c in fa.columns:
            fa[c] = fa[c].apply(to_num)

for gname, cols in FA_GROUPS.items():
    valid = [c for c in cols if c in fa.columns]
    fa[gname] = fa[valid].sum(axis=1, skipna=True) if valid else np.nan

fa_vars = list(FA_GROUPS.keys())
fa_df = fa[["Sample_ID"] + fa_vars].drop_duplicates(subset=["Sample_ID"]).set_index("Sample_ID")

# load sterols

ster = pd.read_excel(STEROL_FILE, sheet_name=STEROL_SHEET)
if "Sample_ID" not in ster.columns:
    raise KeyError("Sterol sheet must contain column 'Sample_ID'.")

ster["Sample_ID"] = ster["Sample_ID"].apply(normalize_id)

STEROL_COLS = [
    "Erg.2 (Dehydroergosterol)",
    "Ergosterol",
    "Stigmastanol",
    "β-Sitosterol",
    "γ-Sitostenone",
]
sterol_vars = [c for c in STEROL_COLS if c in ster.columns]
for c in sterol_vars:
    ster[c] = ster[c].apply(to_num)

ster_df = ster[["Sample_ID"] + sterol_vars].drop_duplicates(subset=["Sample_ID"]).set_index("Sample_ID")


# merge all and filter substrates
merged = org_df.join(fa_df, how="inner").join(ster_df, how="inner")
merged = merged[merged["Substrate_type"].isin(["hardwood", "softwood"])].copy()

if PRINT_DEBUG:
    print("Merged ALL shape:", merged.shape)
```

```
    print("Substrates:", sorted(merged["Substrate_type"].unique()))
    print("Compartments:", sorted(merged["Compartment"].dropna().unique()))

if merged.shape[0] == 0:
    raise ValueError("After filtering to hardwood/softwood, 0 rows left. Check Substrate_type values.")

# panel features
ALL_FEATURES = [c for c in (org_feat_cols + fa_vars + sterol_vars) if c in merged.columns]

panelA_features = [c for c in ALL_FEATURES if c not in PANEL_A_DROP_VARS]

panelB_features = [
    c for c in ALL_FEATURES
    if c not in PANEL_B_DROP_FA_GROUPS
    and c not in PANEL_B_DROP_STEROLS
    and c not in PANEL_B_DROP_ORGANICS
]
panelB_features = [c for c in panelB_features if pretty_label(c) not in {"Lignin", "Cellulose"}]

if PRINT_DEBUG:
    print(f"Panel A features requested: {len(panelA_features)}")
    print(f"Panel B features requested: {len(panelB_features)}")
    print("Panel B dropped FA groups:", PANEL_B_DROP_FA_GROUPS)

# building panels
merged["Comp_Left"] = merged["Compartment"].apply(coarse_compartment)
merged["Comp_Right"] = merged["Compartment"].replace({
    "pileipellis": "Pileipellis",
    "trama": "Trama",
    "hymenium": "Hymenium",
})

panelA = merged.dropna(subset=["Comp_Left"]).copy()
XA = prepare_matrix(panelA, panelA_features, panel_name="Panel A")
subsA = panelA.loc[XA.index, "Substrate_type"].to_numpy()
compsA = panelA.loc[XA.index, "Comp_Left"].to_numpy()

panelB = merged[merged["Comp_Right"].isin(["Pileipellis", "Trama", "Hymenium"])].copy()
XB = prepare_matrix(panelB, panelB_features, panel_name="Panel B")
subsB = panelB.loc[XB.index, "Substrate_type"].to_numpy()
compsB = panelB.loc[XB.index, "Comp_Right"].to_numpy()

if PRINT_DEBUG:
    print("Panel A matrix:", XA.shape)
    print("Panel B matrix:", XB.shape)
    print("Panel B final features used:", list(XB.columns))

if XA.shape[0] < 3 or XB.shape[0] < 3:
    raise ValueError("Too few samples in Panel A or Panel B after filtering / cleaning.")

# PCA panels
def draw_panel(ax, Xdf, subs, comps, marker_map, topk_labels=None):
    X = Xdf.to_numpy()
    feat_names = Xdf.columns.to_numpy()

    pca, scores, load = pca_scores_and_loadings(X)

    ax.set_xlim(scores[:, 0].min() * 1.10, scores[:, 0].max() * 1.10)
    ax.set_ylim(scores[:, 1].min() * 1.10, scores[:, 1].max() * 1.10)

    for sub in ["hardwood", "softwood"]:
        for comp in marker_map:
            sel = (subs == sub) & (comps == comp)
            if np.any(sel):
                ax.scatter(
                    scores[sel, 0], scores[sel, 1],
                    c=color_map[sub],
                    marker=marker_map[comp],
                    s=115,
                    edgecolor="black",
                    linewidths=0.5,
                    alpha=0.9,
                    zorder=3
                )

    xlim = ax.get_xlim()
    ylim = ax.get_ylim()
    radius = 0.45 * min(xlim[1] - xlim[0], ylim[1] - ylim[0])

    strength = np.linalg.norm(load[:, :2], axis=1)
    Lmax = strength.max() if strength.size else 1.0
    scale = radius / Lmax if Lmax > 0 else 1.0

    if topk_labels is None:
        label_idx = set(range(load.shape[0]))
    else:
        k = min(int(topk_labels), load.shape[0])
        label_idx = set(np.argsort(strength)[-k:]) if k > 0 else set()
```

```
    texts = []
    for i in range(load.shape[0]):
        vx, vy = load[i, 0] * scale, load[i, 1] * scale

        ann = ax.annotate(
            "", xy=(vx, vy), xytext=(0, 0),
            arrowprops=dict(
                arrowstyle="-|>",
                lw=ARROW_LW,
                mutation_scale=ARROW_HEAD,
                color="black",
                alpha=ARROW_ALPHA
            ),
            zorder=4
        )
        if ann.arrow_patch is not None:
            ann.arrow_patch.set_clip_on(False)
            ann.arrow_patch.set_path_effects([
                pe.Stroke(linewidth=ARROW_STROKE, foreground="white"),
                pe.Normal()
            ])

        if i in label_idx:
            t = ax.text(
                vx * LABEL_OFFSET, vy * LABEL_OFFSET,
                label_var(feat_names[i]),
                fontsize=LABEL_FONTSIZE,
                ha="center", va="center",
                zorder=6,
                clip_on=False
            )
            t.set_path_effects([
                pe.Stroke(linewidth=LABEL_STROKE, foreground="white"),
                pe.Normal()
            ])
            texts.append(t)

    ax.figure.canvas.draw()
    if texts:
        adjust_text(
            texts,
            ax=ax,
            expand_text=(1.25, 1.25),
            expand_points=(1.15, 1.15),
            force_text=0.8,
            force_points=0.4,
            only_move={"text": "xy"},
            arrowprops=None
        )
        expand_limits_to_texts(ax, texts, pad=LIMIT_PAD)

    ax.axhline(0, color="0.85", lw=1.2, zorder=1)
    ax.axvline(0, color="0.85", lw=1.2, zorder=1)

    ax.set_xlabel(f"PC1 ({pca.explained_variance_ratio_[0]*100:.1f}%)")
    ax.set_ylabel(f"PC2 ({pca.explained_variance_ratio_[1]*100:.1f}%)")

    ax.set_aspect("equal", adjustable="datalim")
    ax.margins(0.06)

    for spine in ax.spines.values():
        spine.set_visible(True)
        spine.set_linewidth(1.0)
        spine.set_color("black")

fig, axes = plt.subplots(
    2, 1,
    figsize=(8, 12),
    dpi=400,
    constrained_layout=True
)

draw_panel(axes[0], XA, subsA, compsA, marker_left, topk_labels=TOPK_LABELS)
draw_panel(axes[1], XB, subsB, compsB, marker_right, topk_labels=TOPK_LABELS)

handles_left = [
    mlines.Line2D([], [], color="black", marker=m, linestyle="None", markersize=10, label=lab)
    for lab, m in marker_left.items()
]
handles_mid = [
    mpatches.Patch(color=color_map["hardwood"], label="Hardwood"),
    mpatches.Patch(color=color_map["softwood"], label="Softwood"),
]
handles_right = [
    mlines.Line2D([], [], color="black", marker=m, linestyle="None", markersize=10, label=lab)
    for lab, m in marker_right.items()
]
```

```
fig.legend(
    handles=handles_mid, title="Substrate type",
    loc="upper center", bbox_to_anchor=(0.5, 1.06),
    frameon=False, ncol=2
)

fig.legend(
    handles=handles_left, title="Compartment",
    loc="upper right", bbox_to_anchor=(1.30, 1.00),
    frameon=True, ncol=1
)

fig.legend(
    handles=handles_right, title="Basidiocarp",
    loc="upper right", bbox_to_anchor=(1.20, 0.50),
    frameon=True, ncol=1
)

plt.show()
```

10 Personal Declaration

Personal declaration: I hereby declare that the submitted thesis is the result of my own, independent work. All external sources are explicitly acknowledged in the thesis.
I acknowledge that ChatGPT (OpenAI) was used to assist with grammar, syntax and content refinement. Furthermore, GPT 5 and Sonnet 4.5 were used to verify and improve the code in its structure, readability, quality and mathematical correctness.

Zurich, April 22, 2026

Ladina Reich