Towards an Uncertainty-Aware BONSAI Database: Bayesian Balancing of Hybrid SUTs

Simon Schulte[1,2], Stefano Merciai[3], Fan Yang[1], Maik Budzinski[1], Bo Weidema[1,3]

[1]Life Cycle Sustainability, Department of Sustainability and Planning, Aalborg University, 9000 Aalborg, Denmark.
[2]Industrial Ecology Freiburg, University of Freiburg, Freiburg, Germany.
[3]2-0 LCA consultants, Aalborg, Denmark

Correspondence: Simon Schulte, simon.schulte@indecol.uni-freiburg.de, Industrial Ecology Freiburg, University of Freiburg, Freiburg, Germany.

Abstract

This technical report documents an initial, working implementation of a Bayesian MCMC framework for balancing BONSAI – a large hybrid Supply-Use Table (HSUTs), developed in the "Getting the Data Right" project.

The aims are twofold: Firstly, to demonstrate feasibility and scalability. The current Stan implementation successfully balances HSUTs with more than 1.8 Million flows on a HPC and produces posterior samples that can in principle be propagated to environmental footprints. The second aims of this report is to make assumptions and simplifications transparent. Several aspects of the model are deliberately simple (e.g. data-driven priors for flows, global uncertainty parameters for transfer coefficients and conversion factors, a basic set of soft constraints). These choices are documented, together with ideas for how they could be refined.

The report and the data in the accompanying Zenodo repository should therefore be read as a proof-of-concept and technical baseline, rather than a final, fully validated methodology.

1 Introduction

1.1 The BONSAI Climate Footprint Database

The BONSAI database is based on an open-source workflow developed in the "Getting the Data Right" project led by Aalborg University and funded by the KR Foundation. BONSAI is a global hybrid Input–Output (IO) system in a make–use (supply–use) framework, where each physical or economic flow is represented in the most appropriate unit: tangible material flows in tonnes, energy flows in terajoules, vehicles and other discrete goods in items, and residual services and value flows in millions of euros.

A core idea of BONSAI is that every transformation from raw statistics to IO-ready data should be explicit, reproducible and inspectable, so that users can understand where numbers come from and how they might be improved. The database produced by this workflow therefore aims to provide not just another global MRIO, but a transparent, updatable and community-extensible hybrid SUT/IOT system that can support robust life-cycle assessment and footprint studies. The current release of BONSAI is available here: www.bonsai.uno.

However, the current BONSAI release provides single point estimates for each flow and consequently supports only deterministic product footprint results. As part of the "Getting the Data Right" project, the work presented here introduces a first fully probabilistic version of BONSAI, based on a Bayesian framework capable of propagating data uncertainty through the hybrid SUT system and generating posterior distributions for final impacts rather than single values. The work presented here therefore forms **a first step toward an uncertainty-aware BONSAI database** that supports more robust interpretation and decision-making.

1.2 Beyond Point Estimates: Why Uncertainty Matters

Uncertainty of Supply-Use Tables (SUTs) matters because these tables underpin virtually all environmentally extended input-output (EEIO) models, and therefore the environmental footprints derived from them. Although SUTs are often treated as if they represent an objective truth about an economy, they are in fact reconciled estimates compiled from heterogeneous, incomplete and sometimes conflicting data sources [23, 14].

When such uncertainty is ignored, the resulting footprints appear more precise than they actually are, giving analysts and decision-makers a false sense of accuracy. This becomes problematic when model results are used to inform real-world decisions - such as comparing technologies, designing carbon taxation, or prioritising decarbonisation pathways - where misleading precision can lead to suboptimal or even harmful policy choices.

In an interconnected global economy, such policies can propagate through trade and supply chains, amplifying their consequences. Quantifying uncertainty enables more robust decision-making by allowing policies to be evaluated not only in terms of expected outcomes, but also in terms of their performance across a range of plausible economic representations [7].

Moreover, transparent uncertainty estimates help identify which assumptions, sectors, or data sources contribute most to overall uncertainty, thereby guiding more efficient model improvement [8]. In this way, acknowledging uncertainty does not weaken the role of SUT-based footprints in policy - it enhances their credibility, interpretability, and practical usefulness for sustainability transitions.

Quantifying the uncertainty of Input-Output (IO) models or SUTs is commonly done by by propagating uncertainty from model input data to model outputs via Monte-Carlo (MC) simulations [21]. This requires not only the uncertainty of the underlying input data used to compile the database (e.g. national SUTs, trade data, production volume data, LCI data, etc.), but also a way to propagate this uncertainty through the construction of the SUT itself to the outputs; in the case of BONSAI, to the footprints. This is non-trivial, because a core of SUT construction is the balancing step where conflicting data sources are reconciled under accounting and other constraints. If balancing is treated as a deterministic pre–processing stage, uncertainty cannot be consistently carried into the final table. This raises the question: how should we balance SUTs when uncertainty is not noise added afterward, but part of the balancing model itself?

1.3 Balancing (hybrid) SUTs

The central part of compiling SUTs is the balancing, in which data conflicts and inconsistencies are resolved to get data that is consistent with mass-balance and accounting identity constraints. While a single monetary unit allows for straightforward balancing within a closed system of equations, hybrid SUTs must reconcile several interlinked constraints: mass, economic and energy balances, along with conversion factors bridging the flows between the different property layers [17].

The most widely applied approach to balance SUTs and IOTs is the biproportional RAS method [24], which iteratively scales rows and columns to match target margins while preserving the initial structure. Alternative optimisation-based approaches minimise a distance measure (such as least squares or cross-entropy) under accounting constraints [26].

1.4 Requirements for uncertainty-consistent balancing

In most IO and SUT applications, balancing and uncertainty analysis are still treated as two separate steps: first, a deterministic balancing procedure (such as RAS) is used to obtain a single "best" table; next, uncertainty is added on top, typically via Monte-Carlo perturbations of that table or of the resulting technical coefficients [13, 28, 11, 18, 22, 30, 10, 1]. In such workflows, the balancing algorithm itself is not part of the probabilistic model but only a pre-processing step.

In this report, however, the balancing procedure is itself the object of inference. This raises the question: what does it mean to treat uncertainty "consistently" in SUT balancing?

For the purposes of this work, we call an uncertainty treatment in the balancing step statistically consistent if it satisfies the following criteria:

1. **Use of heterogeneous reliability information in the balancing step:** Reliability or uncertainty information for individual variables (supply/use flows, conversion factors, transfer coefficients) must be used within the balancing procedure, so that unreliable flows are allowed to move more than reliable flows.
2. **Uncertainty for the balanced table, not only for the raw inputs:** The output of the balancing procedure should not be just a single point estimate, but a probability distribution (or at least a covariance structure) over all balanced flows and parameters.
3. **Accounting for dependence induced by constraints:** The accounting and consistency constraints (product balance, activity balance, consistency across property layers, transfer coefficient consistency) apply across large numbers of cells. A consistent uncertainty treatment must therefore model the dependence structure that these constraints induce, not assume independence of cell-wise errors after balancing.
4. **Support for downstream uncertainty propagation:** The uncertainty description must be rich enough to be propagated to arbitrary downstream quantities of interest (multipliers, footprints, comparisons of products, etc.) including probability statements such as $P(\text{footprint}(X) > \text{footprint}(Y)$.
5. **Validity under nonlinearity and multi-layer constraints:** Methods should remain valid (or at least clearly approximative) when constraints are nonlinear or span multiple property layers (e.g. consistency of the mass, monetary and energy layers, consistency with transfer coefficients).

1.5 Evaluation of methods against requirements

While most of the traditional balancing methods do not consider any uncertainty at all (hence satisfing none of the criteria above), there are few approaches that partly incorportate uncertainty: Weighted least-squares, KRAS, and the Bayesian method of Rodrigues [19]. Those, we will review briefly in the following and assess them against these criteria. This motivates the fully Bayesian MCMC approach developed later (see Section 1.6).

1.5.1 Weighted least-squares

One option to include uncertainty into the balancing is by using weighted least-squares (WLS) optimisation [e.g 29]. WLS searches for a matrix $\mathbf{X}$ that minimises the total *weighted* deviation from the initial estimate $\mathbf{X}_0$ subject to accounting constraints.

The weights represent the inverse variance (or relative reliability) of each cell. Highly reliable flows receive high weights and therefore change little during optimisation; uncertain flows receive low weights and can change more. In this way, uncertainty information can directly be embedded in the balancing process.

From the perspective of the criteria above:

- WLS does incorporate reliability information in the balancing step → satisfies (1).
- However, the optimisation returns a single point estimate only; there is no explicit probabilistic model for the balanced table → fails (2).
- If uncertainty is attached afterwards (e.g. by ad hoc MC around the single point estimate), this is usually done under strong independence and normality assumptions not implied by the balancing optimisation → fails (3).
- Downstream uncertainty propagation is therefore only as credible as those additional assumptions; there is no guarantee that the uncertainty actually reflects the uncertainty of the balancing step → weak (4).
- Nonlinear constraints are often linearised or approximated within the optimisation → at best approximate (5).

In short, WLS methods use uncertainty to weight adjustments but do not provide a coherent joint probability model for the balanced SUT.

1.5.2 KRAS (Lenzen et al.)

KRAS is a generalised iterative scaling method designed for balancing input-output tables and supply-use tables (SUTs) under conflicting external information and inconsistent constraints [12]. It builds on the classic biproportional RAS algorithm and its extension for negative values (GRAS). KRAS adds key features:

- Ability to impose constraints on arbitrarily sized subsets of matrix entries (not just full rows/columns).
- Incorporation of reliability information (standard errors) for both the initial estimate and the external constraints.
- Handles negative values, preserves sign of matrix elements if required.

The algorithm iteratively scales the matrix to satisfy row/column/other constraints while minimising deviation from the initial estimate and respecting reliability weights. When external constraints conflict, KRAS does not force a hard fix of one over the other, but finds a "compromise" solution weighted by the reliability (uncertainty) of data sources.

In the first iteration, KRAS yields one balanced table. In a second iteration, they then use the actual KRAS adjustment as a first guess of SDs and apply a RAS-type scaling on the SDs themselves [13]. Those SDs on the balanced cells can then be used in a MC simulation to propagate uncertainty to the footprints.

Relative to the criteria:

- Reliability information is explicitly used in the iterative scaling → covers (1).
- KRAS does produce SDs for balanced cells and thus nods towards (2), but the SDs are diagnostic, inferred ex post from the realised adjustments rather than derived from a generative probabilistic model.
- Errors in different cells are effectively treated as uncorrelated in the subsequent Monte-Carlo step, despite the strong coupling induced by the constraints → clearly violates (3).
- Downstream uncertainty quantification is possible but inherits the independence assumptions and the specific way SDs were constructed → partial (4).
- Nonlinear and multi-layer constraints can be handled in principle but are not embedded in a joint probability model; their effect on uncertainty is difficult to characterise beyond SDs → weak (5).

Conceptually, KRAS treats balancing as an optimisation problem and then interprets the resulting adjustments as if they were random errors. In a Bayesian setting, the direction is reversed: we start from specified error distributions and constraints, and balanced tables are consequences of that probabilistic model.

1.5.3 Rodrigues (2014)

Rodrigues [19] proposes a Bayesian approach that goes considerably further towards a fully consistent treatment. Rodrigues [19] impose constraints (accounting identities) not only on the first moment (like WLS) but also on second moments (covariance matrix). They find a posterior distribution by relative entropy minimisation (maximum entropy / minimum cross-entropy) subject to these constraints.

In their framework:

- Given uncertainty and quality levels determine which cells move, and by how much. Reliability information is therefore used within the update → satisfies (1).
- The method yields both a posterior mean table and an updated covariance matrix for the balanced flows → directly addresses (2), and crucially does not assume independence → satisfies (3) at the level of second moments.
- Downstream uncertainty propagation is supported via linear error propagation or other moment-based techniques: for linear functionals of the balanced table, the posterior mean and covariance are sufficient → supports (4) for linear or nearly linear summaries.
- However, the posterior is represented only by its first two moments and is derived via linear(ised) updates. There is no explicit sampling, and no information on distributional shape (skewness, heavy tails, multimodality) → (5) holds only locally; and non-Gaussian / strongly nonlinear effects are not captured.

Among the methods reviewed here, Rodrigues (2014) comes closest to the notion of a single coherent probabilistic model for a balanced SUT. Yet it remains limited to Gaussian, moment-based summaries and does not provide explicit posterior samples of the balanced table.

1.6 Bayesian Balancing using Monte-Carlo Markov Chain (MCMC) simulations

Summarizing, none of the approaches reviewed above:

- defines a joint posterior distribution over all flows, conversion factors, transfer coefficients and constraints, from which both a balanced table and its uncertainty are derived,
- captures the full dependence structure induced by multi-layer constraints across different units

- supports (nonlinear) downstream quantities and probability statements without additional approximations.

Here, in this document, we propose a fully Bayesian balancing approach based on Monte-Carlo Markov Chain (MCMC) simulations that addresses these shortcomings by:

1. specifying prior distributions for all unknown flows and parameters (supply, use, conversion factors, transfer coefficients);
2. encoding all accounting and cross-property layer consistency constraints as likelihood terms
3. using MCMC to obtain samples from the joint posterior.

Each posterior draw corresponds to one internally consistent, fully balanced HSUT. The resulting "cloud" of balanced tables carries a coherent description of uncertainty (including correlations) and can be propagated directly to downstream indicators – without the additional independence, linearity or Gaussian assumptions required by the existing approaches.

1.7 Literature review on MCMC-based approaches

There are few studies that use MCMC-based approaches to similar problems, which include (non-extaustively) Cencic and Frühwirth [5], Cencic and Frühwirth [6], Tsionas [25], Lugovoy, Polbin, and Potashnikov [15], Boratyński [3], Wang et al. [27], and Lupton and Allwood [16]. Those are reviewed in more detail in the Appendix (see Section 6.1).

Here, we only summarize the main take-aways from those papers for our problem of balancing BONSAI – a large (>1E6 flows) hybrid SUT:

- **hard vs. soft constraint:** Wang et al. [27] and Boratyński [3] explicitly discuss the advantage of modelling accounting identities (mass balances) as soft constraints (using noise terms) rather as hard constraints, primarily due to computational challenges for sampling a large parameters space with hard constraints (very low acceptance rates). Moreover, Wang et al. [27] provide an elegant interpretation of the noise term on the constraints: "Additionally, the noise term has the interpretation of modeling epistemic or systematic uncertainty in the MFA system. In theory, every process should be exactly mass-conserved if the system definition and diagram are a perfect representation of the underlying real system being modeled. However, in reality, the system diagram and definition are simplified and approximate models of reality, and so it is reasonable to account for uncertainty in the underlying system itself, which can be modeled through a noise term in the mass balance conditions. In practice, flows reported in MFA studies are rarely exactly mass-balanced, so likewise a Bayesian MFA approach does not need to produce perfectly mass-balanced flows to provide useful estimates." [27]
- **size of the problem**: Among all reviewed studies, Tsionas [25] applies MCMC to the largest system (196x196=38416 flows), whilst most others are more in the region of 60-80 flows.
- **computational challenges**: All authors discuss computational challenges when scaling to higher dimensions – in particular those studies using a hard-constraint approach.

1.8 Research Gaps

Bayesian inference using MCMC offers a principled way to obtain full probability distributions for balanced supply-use tables, naturally capturing correlations among flows and enabling uncertainty propagation. However, the literature still lacks a comprehensive Bayesian balancing approach that can integrate both linear and non-linear constraints, operate across multiple physical and monetary

property layers in hybrid SUTs, and scale to system sizes exceeding one million flows, which is several orders of magnitude larger than the case studies reviewed to date.

The implementation described here achieves a first scalable prototype for BONSAI and demonstrates that such a model is computationally feasible. It is intended as a **starting point** for more comprehensive methodological work, rather than a final answer to these research gaps.

2 Methodology

2.1 Bayes Theorem

To obtain a balanced HSUT that is internally consistent across physical and monetary layers (e.g. supply = use, input = output, monetary flow = physical flow × price), we combine prior information from the initial (unbalanced) BONSAI HSUT with system-wide accounting constraints. Bayesian inference provides a principled framework for this integration. Bayes' theorem states that our belief about the unknown variables is updated according to how well they satisfy the constraints:

$$\text{Posterior} \propto \text{Prior} \times \text{Likelihood} \tag{1}$$

- **Posterior**: what we ultimately want – the joint probability distribution of all unknown parameters (supply, use, prices, conversion factors, transfer coefficients) given all data and constraints.
- **Prior**: What we know about individual flows before considering system-wide accounting relationships
- **Likelihood**: In our model this is given by the constraints: The individual data items should match (supply = use, input = output, [EUR] = [price] x [tonnes]), up to a tolerance. If they deviate that outcome is less likely.

However, we cannot compute the posterior in a closed-form way (except for very trivial models).

2.2 MCMC sampling

Yet, the posterior can be approached by constructing a Markov Chain to do Monte-Carlo approximation. On a high-level, any MCMC approximation entails the following steps:

1. Find starting parameter position
2. Propose to jump from that position to somewhere else
3. Evaluate: Is it a "good" place to jump or not? → "good" = probability for the parameter values of the proposed state compared to the current state: `P(accept) = P(proposal) / P(current)`. This leads to regions of high posterior probability being visited relatively more often than those of low posterior probability.
4. Each accepted state is recorded
5. By running enough iterations we will approach the posterior distribution

For step 2 and 3 (proposing a position and evaluating to jump or not), several different sampling algorithms can be used. Sampling algorithms are the rules that tell a Markov chain how to step through the posterior distribution so that collected draws reflect the target uncertainty. Common options include random-walk Metropolis, Gibbs sampling, and Hamiltonian Monte Carlo (HMC), which uses gradients for larger, directed moves (For a more comprehensive collection and disccusion of those, see for example Sanz-Alonso and Al-Ghattas [20]). Here, in this work we use the No-U-Turn

Sampler (NUTS) which is an adaptive variant of HMC that chooses path lengths automatically, giving fast, well-mixed samples with minimal tuning for complex models [9].

2.3 Notation

All variables and parameters that appear in this report are listed in Table 1. Vectors are denoted using bold lowercase letters, matrices using bold uppercase letters. The "true" value of a quantity is denoted as x, observations are noted as y, distribution parameters are denoted using Greek letters (e.g. μ, σ, α, β).

Table 1: Model variables used in the main text. Stan variable names are provided separately in the Appendix Section 6.5.

Symbol	Description	Value/Source
$x_i^{(\text{sup},l)}, x_i^{(\text{use},l)}$	Detailed supply/use flow i in property layer l (tonnes, M€, items, TJ, tonnes-service)	model quantity: Equation 2; Equation 3
$\mu_i^{(\text{sup},l)}$, $\mu_i^{(\text{use},l)}$	Prior log-scale "best guess" for each supply/use flow	initial BONSAI HSUT
$\sigma_i^{(\text{sup},l)}$, $\sigma_i^{(\text{use},l)}$	Prior log-scale uncertainty (sd) for each supply/use flow	see Section 2.5.1
$x_i^{(cf,l_1\to l_2)}$	Conversion factor for product i from from property layer l_1 to property layer l_2 (e.g., €/t, t/item, TJ/t, €/TJ)	model quantity: Equation 4
$y_i^{(cf,l_1\to l_2)}$	Observed conversion factor value	upstream BONSAI pipeline
$\sigma_i^{(cf,l_1\to l_2)}$	Measurement error (sd) for $y^{(cf,l_1\to l_2)}$	see Section 2.5.2
$\text{CV}^{(\text{tc},l_1\to l_2)}$	Global coefficient of variation per conversion factor ($l_1 \to l_2$) used to compute $\sigma_i^{(cf,l_1\to l_2)}$	0.5 (Section 2.5.2)
$x_i^{(\text{tc})}$	Transfer coefficient i (fraction linking use to subsequent supply)	model quantity: Equation 5
$\alpha^{(\text{tc})}$, $\beta^{(\text{tc})}$	Global Beta prior shape parameters for all $x_i^{(\text{tc})}$	1.0; 1.0 (Section 2.5.2
$y_i^{(\text{tc})}$	Observed value for transfer coefficient n	upstream BONSAI pipeline
$\sigma_i^{(\text{tc})}$	Measurement error (sd) for $y_i^{(\text{tc})}$	see Section 2.5.2
$\text{CV}^{(\text{tc})}$	Global coefficient of variation used to compute $\sigma_i^{(\text{tc})}$ for transfer coefficients	0.1 (Section 2.5.2
$A_{\text{pr}}^{(l,\text{sup})}$, $A_{\text{pr}}^{(l,\text{use})}$	Aggregation matrices: detailed supply/use → product-region totals	constructed from initial BONSAI HSUT
$A_{\text{ar}}^{(l,\text{sup})}$, $A_{\text{ar}}^{(l,\text{use})}$	Aggregation matrices: detailed supply/use → activity-region totals	constructed from initial BONSAI HSUT
$\mathbf{s}_{\text{pr}}^{(l)}$, $\mathbf{u}_{\text{pr}}^{(l)}$	Aggregated supply/use by product-region in property layer l	Equation 8
$\mathbf{o}_{\text{ar}}^{(l)}$, $\mathbf{i}_{\text{ar}}^{(l)}$	Aggregated output/input by activity-region in property layer l	Equation 9
$\text{rel\_tol}_{\text{product}}$	Allowed log-ratio deviation for product balances	0.05 (Section 2.4.3.1)

Symbol	Description	Value/Source
$\text{rel\_tol}_{\text{activity}}$	Allowed log-ratio deviation for activity balances	0.05 (Section 2.4.3.1)
$\text{rel\_tol}_{\text{conv\_fac}}$	Allowed log-ratio deviation for conversion factor consistency	0.1 (Section 2.4.3.2)
$\text{rel\_tol}_{\text{tc}}$	Allowed log-ratio deviation for transfer coefficient consistency	0.2 (Section 2.4.3.3)
$sd_{\min}, ; sd_{\max}$	Lower and upper bounds used to clamp log-scale standard deviations $\sigma_i^{(\text{sup},l)}, \sigma_i^{(\text{use},l)}$ to a conservative range	0.1; 0.5 (Section 2.5.1
ϵ	Small positive constant to stabilise log ratios	0.001 (Section 2.4.3)

2.4 The model

This section provides a conceptual overview of the probabilistic model used to balance BONSAI. The goal is to explain what the model infers, what information it uses, and how constraints are enforced. Readers interested in full implementation details may refer to the Stan code in the Appendix (Section 6.6) and the mapping between the mathematical formulations shown in the following and the Stan code (Section 6.5).

At a high level, the balanced HSUT is obtained by treating all flows and linking parameters as *unknown random variables* (Section 2.4.1), assigning them *prior distributions* based on available data (Section 2.4.2), and updating these beliefs using *accounting constraints* expressed as likelihood terms (Section 2.4.3). Markov Chain Monte Carlo (MCMC) sampling is then used to draw from the posterior distribution (Section 2.4.4), giving not a single table but a distribution over many possible balanced tables.

2.4.1 Unknown quantities treated as random variables

Instead of assuming the elements of the HSUT as fixed, we treat the following variables as *unknown but estimable*:

- **Supply flows:** $x_i^{(\text{sup},l)} \geq 0$ for property layer $l \in \{\text{tonnes}, \text{Euro}, \text{items}, \text{TJ}, \text{tonnes} - \text{service}\}$
- **Use flows:** $x_i^{(\text{use},l)} \geq 0$
- **Conversion factors / prices:** $x_i^{(\text{cf},l_1 \to l_2)}$ (€/t, t/item, TJ/t, €/TJ)
- **Transfer coefficients:** $x_i^{(\text{tc})}$ (fraction of a given input ending up embodied in a specific output), each $0 < x_i^{(\text{tc})} < 1$

A conceptual distinction is made between types of quantities:

- Supply and use flows are treated as latent "true" quantities for which the initial BONSAI values provide prior information but are not considered exact observations.
- Conversion factors and transfer coefficients are treated as observed values with measurement error, meaning the BONSAI point values act as noisy data that inform – but do not strictly determine – the model's estimates.

This choice is pragmatic rather than theoretically mandatory. For supply/use flows in particular, using the initial BONSAI values as prior means on latent flows is mathematically equivalent to treating them as observations with measurement error of the same variance, and the resulting

posterior would be unchanged. Likewise, transfer coefficients and conversion factors could also be modelled as latent quantities with priors instead of noisy observations. Alternative formulations may be preferable depending on the desired interpretation (prior vs. measurement). Section 4.2.2 discusses this modelling assumption and its implications.

2.4.2 Prior distributions from initial BONSAI data

All uncertain quantities require prior distributions. Because supply and use flows are strictly positive and vary over orders of magnitude, we assign them lognormal priors centred on the initial BONSAI estimates.

Conversion factors and transfer coefficients, in contrast, are considered observations subject to measurement error. We therefore model them as normally distributed around the values provided by the upstream BONSAI pipeline, with uncertainty reflected by a standard deviation parameter (and truncated to $[0, \text{inf}]$ or $[0, 1]$ where necessary).

The process of assigning prior uncertainties to the supply/use flows and measurement errors to the conversion factors and transfer coefficients are outlined further below (Section 2.5).

2.4.2.1 Supply and Use flows

For each flow (supply and use) we require:

- a best estimate (in logs) μ_i (from the initial BONSAI HSUT)
- and an uncertainty range σ_i (Section 2.5)

We encode this as:

$$\log x_i^{(\text{sup},l)} \sim \mathcal{N}(\mu_i^{(\text{sup},l)},\ \sigma_i^{(\text{sup},l)\,2}) \tag{2}$$

$$\log x_i^{(\text{use},l)} \sim \mathcal{N}(\mu_i^{(\text{use},l)},\ \sigma_i^{(\text{use},l)\,2}) \tag{3}$$

This approach reflects the assumption that the initial BONSAI HSUT estimates represents our best guess of the true flows, but with uncertainty, and the model may update them to satisfy accounting constraints and property layer consistency.

2.4.2.2 Conversion factors with measurement error

Conversion factors are treated as observed quantities with noise, meaning the conversion factors provided by the upstream BONSAI pipeline ($y_i^{(cf,l_1\to l_2)}$) are taken as measurements rather than prior beliefs. The latent quantity ($x_i^{(cf,l_1\to l_2)}$) represents the true (unknown) conversion factor:

$$x_i^{(cf,l_1\to l_2)} \sim \mathcal{N}(y_i^{(cf,l_1\to l_2)},\ \sigma_i^{(cf,l_1\to l_2)2}) \tag{4}$$

2.4.2.3 Transfer coefficients: expected value + observation error

Transfer coefficients represent fractions of an input (physically) embodied in a particular output. They are modelled using:

1. a *prior* constraining values to $(0, 1)$, expressed through a Beta distribution,
2. a *likelihood* that treats the BONSAI value ($y_i^{(\text{tc})}$) as a noisy observation:

$$x_i^{(\mathrm{tc})} \sim \mathrm{Beta}(\alpha_i^{(\mathrm{tc})},\ \beta_i^{(\mathrm{tc})}) \tag{5}$$

$$y_i^{(\mathrm{tc})} \sim \mathcal{N}(x_i^{(\mathrm{tc})},\ \sigma_i^{(\mathrm{tc})2}) \tag{6}$$

2.4.3 Representing constraints as likelihood terms

We include the following constraints in our model:

- product balances (for each product and property layer: total use should equal total supply)
- activity balances (for each activity and property layer: total inputs should equal total outputs)
- consistency across property layers (€ ↔ TJ ↔ t ↔ items, via cell-wise conversion factors)
- transfer coefficient consistency (how much of a given inputs ends up embodied in a specific output)

Since several studies reviewed above and in the Appendix mention computational problems with posing hard constraints on the data, particularly in high-dimensions (see Section 1.7 and Section 6.1), in our model perfect equality is *not* forced. Instead, following the framework proposed by Wang et al. [27], we model these accounting identities as soft constraints by including a stochastic noise term. This choice serves two critical functions:

1. Computational Feasibility: It avoids the extreme difficulty of sampling from a "manifold of measure zero" (where the probability mass exists only on a extremely thin line of perfect equality), which causes poor mixing in MCMC samplers.
2. Epistemic Realism: As Wang et al. [27] argue, the noise term allows the model to capture "epistemic or systematic uncertainty." Since any system definition is a simplified approximation of reality, and reported data is rarely perfectly balanced in practice, enforcing strict equality would imply a precision that does not exist. The soft constraint acknowledges this structural uncertainty.

In formal terms, we model the constraints as log-ratios centered on zero, allowing small deviations determined by a tolerance parameter:

$$\log \frac{x_i + \epsilon}{x_j + \epsilon} \sim \mathcal{N}(0,\ \mathrm{rel\_tol}^2), \tag{7}$$

where:

- x_i and x_j are the two quantities that should be balanced (e.g. total use and total supply of a product)
- ϵ is a small positive constant to stabilise log ratios
- rel_tol is the allowed log-ratio deviation from the balance

The rel_tol parameter is crucial from a computational perspective: Shrinking rel_tol tightens the constraints and makes it harder to reconcile them with the prior when imbalances are large (see discussion in Section 4.3). For the model run reported here, we chose constraint tolerances ($\mathrm{rel\_tol}_*$) that allow product and activity balances to deviate by roughly ±10% (95% interval, rel_tol = 0.05), and conversion-factor and transfer-coefficient constraints by about ±20% (rel_tol = 0.1) and ±40% (rel_tol = 0.2) respectively. These values are a compromise: tighter tolerances led to poor mixing and frequent divergences, whereas the reported settings still enforce meaningful approximate balance while keeping the posterior geometry tractable for NUTS.

2.4.3.1 Product and activity constraints

Before the product and activity constraints can be expressed, individual flows are aggregated to product-region and activity-region totals. For example, all Danish coal use-flows are summed to one figure, "total use of coal from Denmark". Likewise, all inputs into coke production in Germany are summed into one figure "total input into coke production in Germany". This aggregation is implemented using sparse aggregation (or mapping) matrices $\boldsymbol{A}$:

$$\mathbf{s}_{\text{pr}}^{(l)} = A_{\text{pr}}^{(l,\text{sup})}\,\mathbf{x}^{(\text{sup},l)}, \qquad \mathbf{u}_{\text{pr}}^{(l)} = A_{\text{pr}}^{(l,\text{use})}\,\mathbf{x}^{(\text{use},l)} \tag{8}$$

$$\mathbf{o}_{\text{ar}}^{(l)} = A_{\text{ar}}^{(l,\text{sup})}\,\mathbf{x}^{(\text{sup},l)}, \qquad \mathbf{i}_{\text{ar}}^{(l)} = A_{\text{ar}}^{(l,\text{use})}\,\mathbf{x}^{(\text{use},l)} \tag{9}$$

Where:

- $\mathbf{s}_{\text{pr}}^{(l)}$: supply by product-region (vector)
- $\mathbf{u}_{\text{pr}}^{(l)}$: use by product-region (vector)
- $\mathbf{o}_{\text{ar}}^{(l)}$: output by activity-region (vector)
- $\mathbf{i}_{\text{ar}}^{(l)}$: input by activity-region (vector)
- $\mathbf{A}_{\text{pr}}^{(l,\text{sup})}$: matrix aggregating supply by product-region
- $\mathbf{A}_{\text{pr}}^{(l,\text{use})}$: matrix aggregating use by product-region
- $\mathbf{A}_{\text{ar}}^{(l,\text{sup})}$: matrix aggregating supply by activity-region
- $\mathbf{A}_{\text{ar}}^{(l,\text{use})}$: matrix aggregating supply by activity-region

For each product-region j (e.g. coal from Denmark) and property layer l we ensure that supply should approximately equal use, with a small tolerance level. We model this via:

$$\log \frac{s_{\text{pr},j}^{(l)} + \epsilon}{l_{\text{pr},j}^{(l)} + \epsilon} \sim \mathcal{N}(0,\ \text{rel\_tol}^2_{\text{product}}) \tag{10}$$

Exactly the same logic applies for each activity-region k (e.g. coke production in Germany):

$$\log \frac{o_{\text{ar},k}^{(l)} + \epsilon}{i_{\text{ar},k}^{(l)} + \epsilon} \sim \mathcal{N}(0,\ \text{rel\_tol}^2_{\text{activity}}) \tag{11}$$

This enforces approximate balancing of the total inputs and outputs of an activity-region pair across all units.

2.4.3.2 Conversion factor consistency across layers

Conversion factors link flows recorded in different property layers. For each flow observation where a conversion factor exists, the corresponding flow in the *source property layer* (l_1) is converted into a *target property layer* (l_2) via:

$$x_i^{(cf,l_1\to l_2)} :\ l_1 \to l_2 \tag{12}$$

$$\underbrace{x_i^{(\text{sup},l_2)}}_{\text{reference flow in } l_2} \approx \underbrace{x_i^{(\text{sup},l_1)} \cdot x_i^{(cf,l_1 \to l_2)}}_{\text{converted from } l_1} \tag{13}$$

More formally, for each applicable conversion factor i we calculate:

$$\text{converted}_i^{(\text{sup},l_2)} = x_i^{(\text{sup},l_1)} \cdot x_i^{(cf,l_1 \to l_2)}, \qquad \text{reference}_i^{(\text{sup},l_2)} = x_i^{(\text{sup},l_2)} \tag{14}$$

(and analogously for use flows $x^{(\text{use},l)}$).

To enforce consistency across property layer, we again apply a soft log-ratio constraint:

$$\log \frac{\text{converted}_i^{(\cdot)} + \epsilon}{\text{reference}_i^{(\cdot)} + \epsilon} \sim \mathcal{N}\Big(0,\ \text{rel\_tol}^2_{\text{conv\_fac}}\Big) \tag{15}$$

2.4.3.3 Transfer-coefficient consistency

Transfer coefficients $x_i^{(\text{tc})}$ describe how much of a given *input* into an activity ends up embodied in a specific *output*. For example, for the activity "car manufacturing" we might have a transfer coefficient of 0.95 for a specific input (e.g. steel) and a specific output (e.g. a car) stating that around 95% of the steel input is *embodied* in the car. In the model, transfer coefficients are checked for consistency against observed supplies in tonnes, conditional on the use flows that feed into them.

1. First, we have to aggregate supply and use to the product-activity-region (PAR) level (e.g. total use of steel in car manufacturing in Germany, or supply of cars from car manufacturing in Germany) using aggregation matrices $\mathbf{A}_{par}^{(\cdot)}$:

$$\mathbf{x}_{\text{par}}^{(\text{use,tonnes})} = \mathbf{A}_{\text{par}}^{(\text{use,tonnes})}\, \mathbf{x}^{(\text{use,tonnes})} \tag{16}$$

$$\mathbf{x}_{\text{par}}^{(\text{sup,tonnes})} = \mathbf{A}_{\text{par}}^{(\text{sup,tonnes})}\, \mathbf{x}^{(\text{sup,tonnes})} \tag{17}$$

2. Second, we compute the expected supply from the transfer coefficients by multiplying the transfer coefficient record with the respective use-flow at the PAR-level:

$$x_{\text{par},i}^{(\text{exp\_sup,tonnes})} = x_i^{(\text{tc})} \cdot x_{\text{par},i}^{(\text{use,tonnes})} \tag{18}$$

3. Third, we again apply a soft log-ratio constraint so that the observed PAR-level supplies in tonnes must approximately match expected supplies implied via transfer coefficients:

$$\log \frac{x_{\text{par},i}^{(\text{exp\_sup,tonnes})} + \epsilon}{x_{\text{par},i}^{(\text{sup,tonnes})} + \epsilon} \sim \mathcal{N}\left(0, \text{rel\_tol}^2_{\text{tc}}\right) \tag{19}$$

2.4.3.4 Technical note: Applying the constraints to subsets of the data

All four constraints (activity & product balance, transfer coefficient and conversion factor constraints) are not applied to *all* supply/use flows but only to a *subset*:

- The product constraint is applied to commodities (`C_`), market products (`M_`) and energy products (`EF_`). Products being produced but not used or vice versa are excluded (e.g. resource input or emission output).
- The activity constraint is applied to all activities `A_` and markets `M_`, excluding final consumption, markets of fuel mixes, service activities (= activities with physical inputs but only monetary outputs). Activities without inputs/outputs are excluded.
- The conversion factor constraint is applied to those flows which are recorded at least in two units and have a conversion factor.
- The transfer coefficient constraint is applied to those flows for which a transfer coefficient is provided.

2.4.4 Posterior inference using MCMC

We implemented the model described above in Stan – a probabilistic programming language for describing probabilistic models and inferring the posterior via MCMC simulations. Given priors (Section 2.5) and likelihoods (Section 2.4.3), Stan uses the No-U-Turn Sampler (NUTS) to draw samples from the joint posterior distribution of all flows and parameters.

At a high level, MCMC sampling involves the following steps:

1. Propose a candidate balanced SUT (all flows, conversion factors, transfer coefficients).
2. Evaluate how well it fits:
   - prior information and measurement errors (Section 2.5),
   - SUT balance constraints (Section 2.4.3).
3. Accepts or rejects the proposal based on overall fit.
4. By repeating step 1-3 hundreds of times it produces a **cloud of balanced SUTs**.

From this cloud one can compute:

- A central balanced SUT (mean)
- Uncertainty ranges for each cell
- Covariances between individual cells

Moreover, and more importantly, this cloud of plausible balanced SUTs can be used in a subsequent MC simulation to propagate the uncertainty further to the final footprints.

2.5 Specifying prior uncertainty

The balancing model as described above requires two parameters for every flow: a best-guess (μ) and an uncertainty estimate (σ). While μ is provided by the initial BONSAI point estimates, specifying σ is challenging because the raw data used to compile the initial BONSAI HSUT does not include granular uncertainty metadata (e.g. in form of standard errors per data point).

In the course of our work on this subject, we made the experience that in such a high-dimensional system ($> 10^6$ parameters) subject to several strict constraints, arbitrarily fixing σ made it impossible for the MCMC sampler to traverse the posterior. If priors are set too narrow (overconfident), the solution space may become disjoint or non-existent – a situation where no configuration of flows can

simultaneously satisfy the "pull" of the priors and the "push" of the constraints. If priors are set too wide, the posterior becomes weakly identified resulting in diverging chains.

To ensure computational feasibility and stable inference, the log-scale standard deviations σ are not fixed arbitrarily but calibrated from the data itself in a simple Empirical-Bayes inspired manner [4] – specifically, from the observed supply–use and activity imbalances in the initial table.

2.5.1 Supply and Use flows

For supply and use flows, we assume that the initial mismatch between supply and use provides a proxy for the "noise floor" of the data. If the reported supply and use for a product differ considerably, it implies that the underlying data sources are noisy. Conversely, a near-perfect balance implies higher precision.

We derive the prior scale variances $\sigma_i^{(sup,l)}$ and $\sigma_i^{(use,l)}$ using the following logic:

1. **Quantify the Imbalance:** We compute the log-difference Δ for each accounting balance (product or activity) at the product-region level (pr) and activity region level (ar), respectively:

$$\Delta_{\text{pr},j}^{(l)} = \log(s_{\text{pr},j}^{(l)} + \epsilon) - \log(u_{\text{pr},j}^{(l)} + \epsilon) \tag{20}$$

$$\Delta_{\text{ar},k}^{(l)} = \log(i_{\text{ar},k}^{(l)} + \epsilon) - \log(o_{\text{ar},k}^{(l)} + \epsilon), \tag{21}$$

   where $s_{\text{pr},j}^{(l)}$ and $u_{\text{pr},j}^{(l)}$ is the total supply/use by product-region j (e.g. coal from Denmark) and property layer l; and $i_{\text{ar},k}^{(l)}$ and $o_{\text{ar},k}^{(l)}$ is the total input/output by activity-region k (e.g. coke production in Germany) and property layer l.
2. **Derive the Noise Level:** Next, to each of these observed imbalances $\Delta_{\text{pr},j}^{(l)}$ and $\Delta_{\text{ar},k}^{(l)}$ there are $m_{\text{pr},j}^{(l)}/m_{\text{ar},k}^{(l)}$ contributing flows. For instance, to the imbalance $\Delta_{\text{pr,German\_steel}}^{(\text{tonnes})}$ the contributing flows are "use of steel by German car manufacturing", "use of steel by German machinery manufacturing" and "supply of steel by German steel manufacturing", amongst others. We assume that these observed imbalances $\Delta_{(\cdot)}$ each sit at the edge of a 95% confidence interval generated by the sum of m noisy rows. This allows us to back-calculate the required σ's for each contributing flow:

$$\sigma_i^{(\text{pr},l)} \approx \frac{|\Delta_{\text{pr},j}^{(l)}|}{1.96 \cdot \sqrt{2} \cdot \sqrt{m_j}}, \tag{22}$$

$$\sigma_i^{(\text{ar},l)} \approx \frac{|\Delta_{\text{ar},k}^{(l)}|}{1.96 \cdot \sqrt{2} \cdot \sqrt{m_k}}, \tag{23}$$

   where $\sigma_i^{(\text{pr},l)}/\sigma_i^{(\text{ar},l)}$ are the flow-specific variances.
   Here, the $\sqrt{m}$ term accounts for the Central Limit Theorem: as more flows contribute to a balance, individual flow uncertainties contribute less to the aggregate uncertainty.
3. For all flows that are subject to more than one constraint, take the largest relevant $\sigma_i = \max\{\sigma_i^{(\text{pr},l)}, \sigma_i^{(\text{ar},l)}\}$ (most conservative) and pass it to the model as the log-scale prior for that flow.
4. **Clamping:** To prevent the priors from becoming physically unrealistic (too wide) or numerically unstable (too narrow), the derived σ_i is clamped to a conservative range $[sd_{\min}, sd_{\max}]$.

We found that this approach improves the geometry of the posterior distribution so that it is sufficiently well-behaved for the NUTS sampler to explore it efficiently, avoiding pathological behavior such as extremely small step sizes and divergent transitions. This provides a robust baseline for the "Proof of Concept" phase.

2.5.2 Transfer coefficients and Conversion factors

Unlike supply and use flows, transfer coefficients and conversion factors are modelled as "observed quantities with measurement error" rather than latent parameters.

In the current prototype, the uncertainty for these parameters is handled via global error scales:

- **Transfer Coefficients:** A single global coefficient of variation $\mathrm{CV}^{(tc)}$ is used to calculate standard deviations $\sigma_i^{(tc)}$ per transfer coefficient.
- **Conversion Factors:** One global coefficient of variation $\mathrm{CV}^{(tc,l_1\to l_2)}$ per conversion-factor layer is used to calculate standard deviations $\sigma_i^{(tc,l_1\to l_2)}$ per layer type.

While this is a simplification, the model architecture already supports flow-specific σ values. Future iterations will replace these global parameters by flow-specific σ's or with hierarchical priors, allowing the model to "learn" different uncertainty levels for different sectors or regions based on the coherence of the data.

2.6 Computational requirements

The current implementation has been tested on the BONSAI HSUT with approximately 1700 products and 1500 activities, each in 49 countries/regions – giving a total of more than 1.8 Million non-zero flows across four property layers (tonnes, Million Euro, TJ, items). On the High-Performance Cluster (HPC) of the University of Aalborg, a typical run with 4 parallel chains and 500-1000 iterations per chain required on the order of 19 to 44 hours and 24 GB of RAM, indicating that full Bayesian balancing at BONSAI scale is practically feasible.

Yet, processing the Stan output files with the current implementation requires around 200 GB of RAM. However, there are several possible options to decrease memory utilisation by switching to more optimized file formats such as HDF5.

3 Results

3.1 Validation: Convergence of chains

For the here-presented model configuration (see Table 1 for parameter values), overall convergence diagnostics are encouraging. Sampling completed without divergences and without transitions hitting the maximum tree depth. 99.98% of all variables have an $\hat{R}$ below 1.05, indicating good mixing and agreement between chains (Figure 1, left). The remaining 0.02% of the variables with an $\hat{R} > 1.05$ (Figure 1, right) point to parts of the system where model structure, priors or constraints could be refined in future work (see Figure 12 in the Appendix for traceplots of the 16 variables with the hightest $\hat{R}$).

Figure 2 shows trace plots (top row) and marginal posterior densities (bottom row) for three exemplary variables/parameters in the Chinese iron & steel sector: the fraction of iron embodied in steel (transfer coefficient), the price of steel supplied (Million €/t), and the physical steel supply (t). The four MCMC chains overlap closely, exhibit no visible trends or drifts, and yield nearly

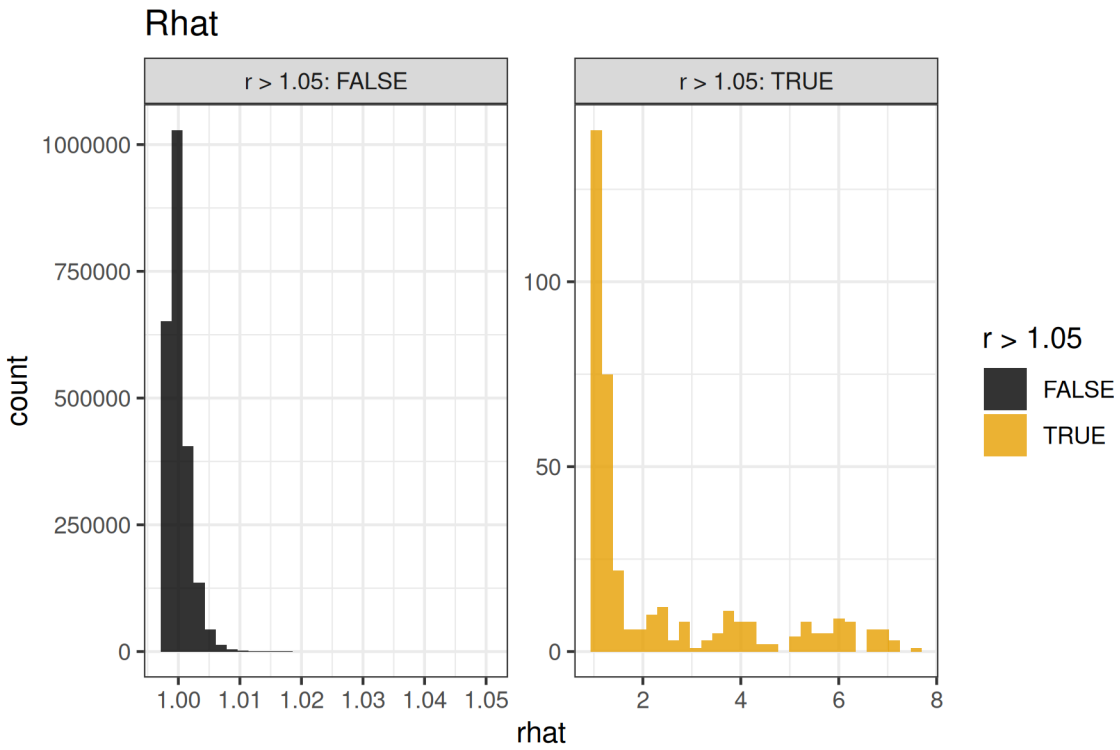


Figure 1: Rhat diagnostics: The $\hat{R}$ statistic compares between-chain and within-chain variance. Values close to 1 signal that independent chains are exploring the same posterior region and have effectively converged.

identical marginal distributions for each parameter, illustrating good mixing and convergence for these examples. Traceplots of a random selection of 16 variables can be found in the Appendix (Figure 11).

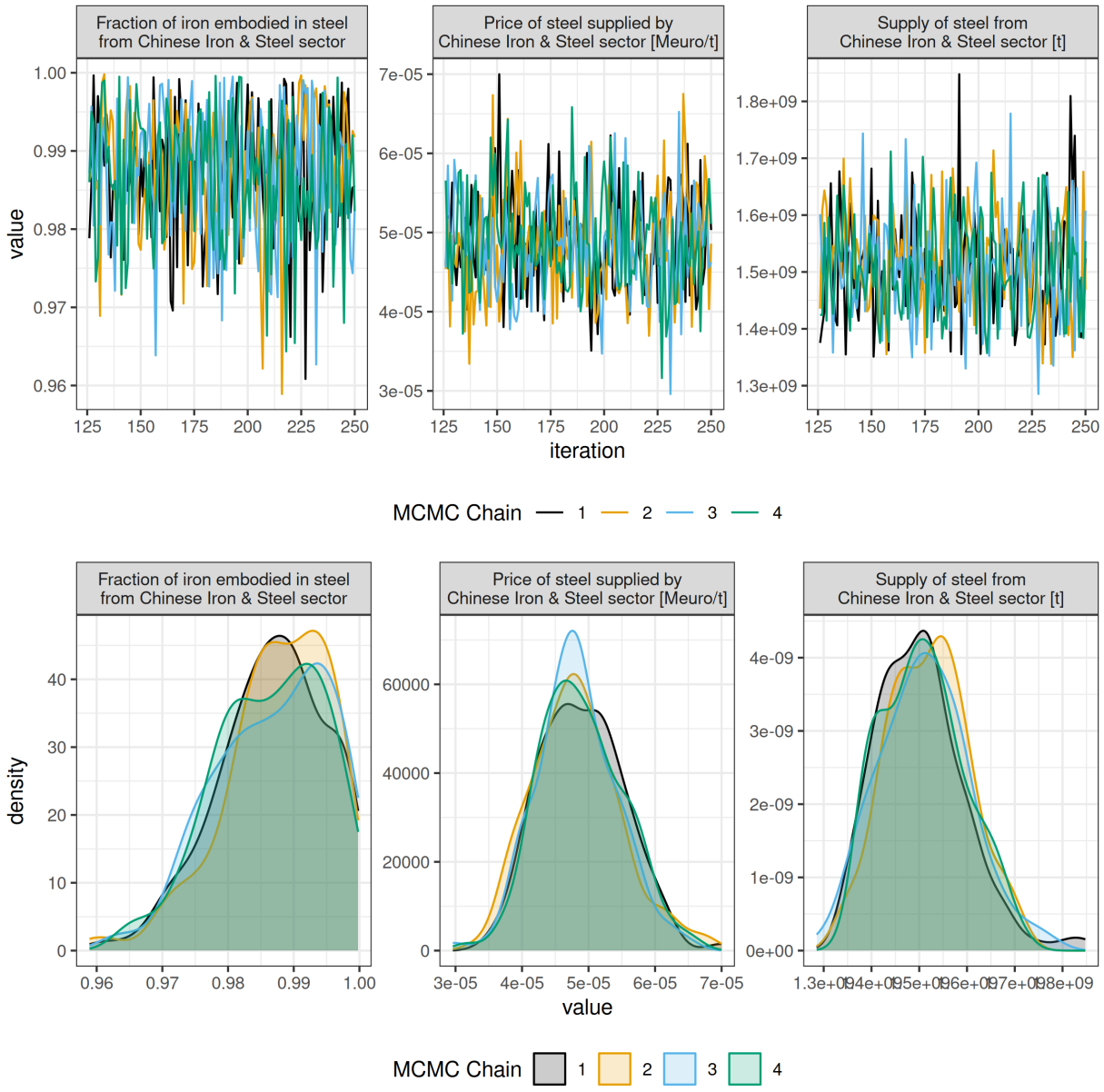


Figure 2: Representative trace (top row) and density plots (bottom row) for selected flows and parameters.

3.2 How much did balancing change the table?

Next, we investigate how the initial supply and use flows have changed in the balancing.

3.2.1 Prior vs. Posterior distributions

First, we compare the initital BONSAI values (prior) with the mean of the posterior distribution. Figure 3 shows hexbin scatter plots of the log posterior mean against the log prior (BONSAI) value for all supply and use flows, respectively. The colour scale indicates the number of flows in each bin, and the red 45 degree line marks exact agreement between prior and posterior. Most mass lies close to this line, indicating most of the initial BONSAI values are only adjusted slightly in the balancing, while the visible spread around the diagonal – more pronounced for use flows than for supply flow – highlights those parts of the system where the balancing constraints induce substantial adjustments.

Figure 4 compares prior and posterior coefficients of variation (CVs) for all supply and use flows on a log scale. The histogram in the left panel shows that the prior CVs (orange) are almost entirely concentrated in a narrow spike around 0.1, with a very thin tail towards much larger values – a direct consequence of the prior sigma construction that assigns a common baseline uncertainty to most flows and inflates it only where needed to explain large imbalances (Section 2.5). The posterior CVs (grey) remain in roughly the same range but are more dispersed around the spike, indicating that the model adjusts uncertainty up or down depending on how strongly each flow is constrained. The ECDF in the right panel confirms this: the prior curve jump sharply near 0.1, while the posterior curve rises slightly earlier and more smoothly, implying that a larger share of flows have CVs just below 0.1 and that extreme CVs become even rarer after balancing.

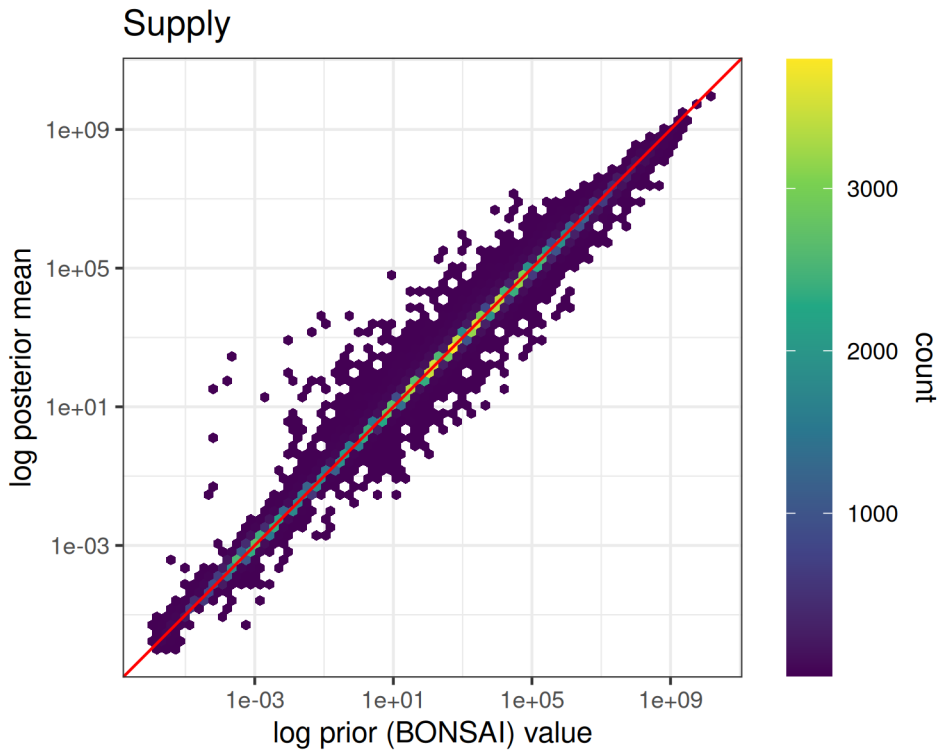
3.2.2 Before vs after balance for key constraints

Figure 5 illustrates how the Bayesian balancing procedure reduces product-level imbalances across the different property layers. Each panel shows hexbin scatterplots of total supply versus total use for all products in the mass (tonnes), monetary (M€), and energy (TJ) layers, before (left) and after (right) balancing. Before balancing, many products deviate substantially from the red 45 degree line (exact equality), indicating considerable inconsistencies between supply and use. After balancing, almost all points concentrate tightly around the diagonal in all three layers, demonstrating that the model brings product balances close to equality while still allowing for small residual deviations due to the soft-constraint specification.

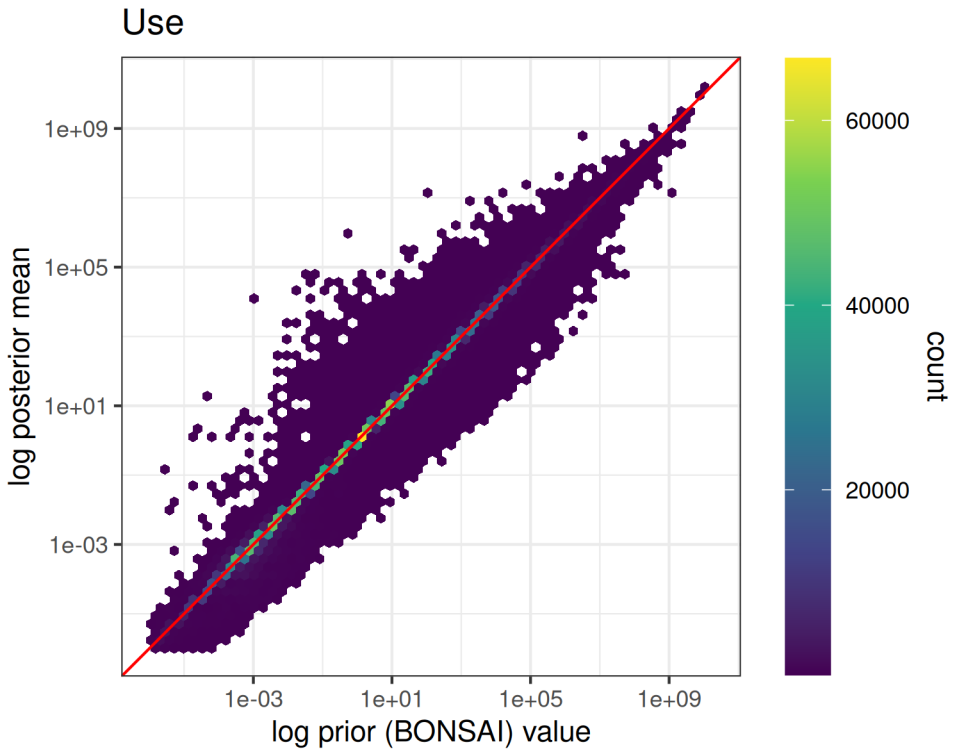
Similarly in case of the activity balance: Prior to balancing, many activities exhibit inconsistencies between total inputs and outputs per activity – although less pronounced than for the product balance. After balancing, the points align closely along the diagonal in all three layers, demonstrating that the model effectively reconciles activity balances.

3.3 Posterior correlations between linked flows: Chinese coke example

A key feature of here-proposed MCMC balancing is that the posterior distribution reflects dependencies between variables resulting from the constraints. As an example, Figure 7 illustrates the posterior dependence between the supply of Chinese coke and its main downstream uses in the balanced HSUT. The diagonal panels show the marginal posterior distributions for each flow, while the lower triangle displays pairwise scatterplots with fitted regression lines and the upper triangle reports the corresponding posterior correlation coefficients. The supply of coke from the Chinese market activity (M_COKE) exhibits moderate positive correlations with coke use by natural coke production (A_Nat_COKE, $\rho = 0.40$) and by steel-making (A_STEL, $\rho = 0.44$), indicating that these major downstream users move jointly with total supply in the posterior. In contrast, correlations with other uses remain close to zero, and the correlation between coke use in natural coke production and steel-making is slightly negative ($\rho = -0.16$), suggesting a weak competitive



(a)



(b)

Figure 3: Hexbin scatterplot of the log posterior mean against the log prior value (from the initial BONSAI table). The colour scale indicates the number of flows in each bin. The red line marks exact agreement between the two.

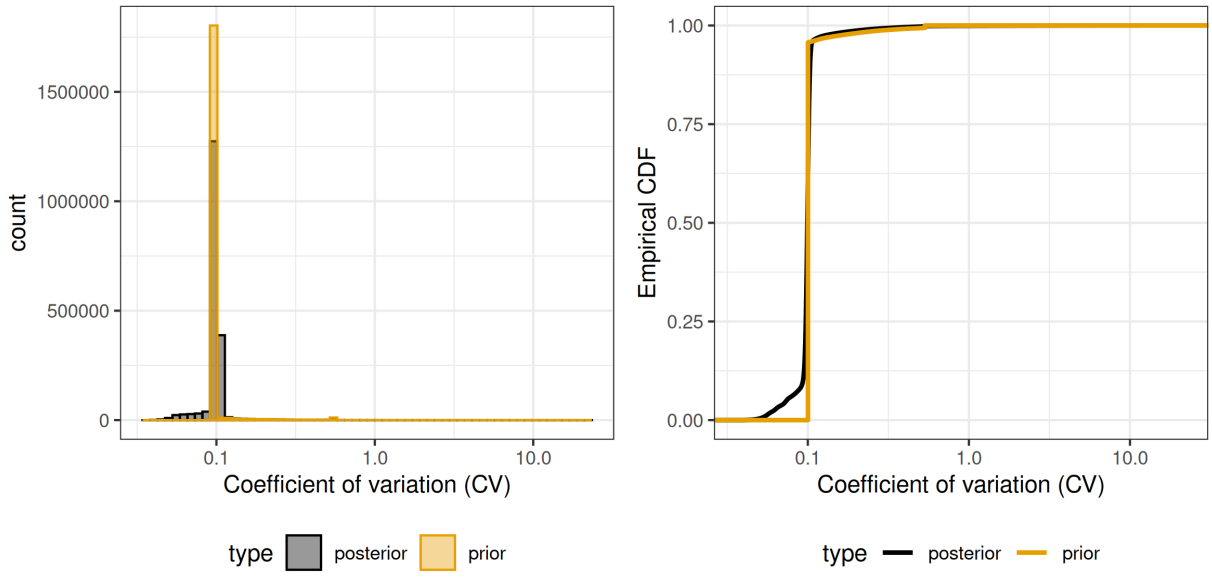
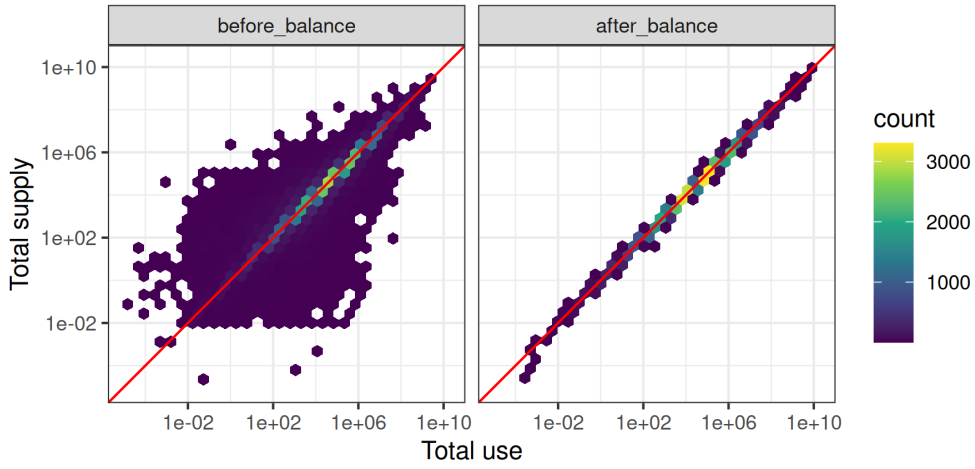


Figure 4: Histograms (left) and empirical cumulative distribution functions (right) of the coefficients of variation (CVs, log scale) for all supply and use flows, comparing prior (yellow) and posterior (black) uncertainty. In the ECDF, the y-value at a given CV on the x-axis indicates the share of flows with CV less than or equal to that value.

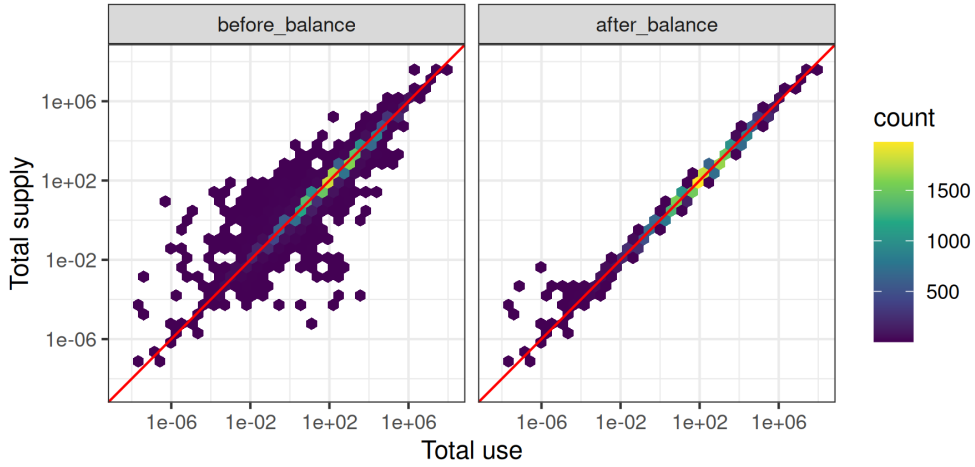
relationship between these two demand channels. Overall, the figure illustrates how the balancing constraints induce a moderate pattern of dependencies between physically linked flows that were a priori treated as independent. Due to use of soft constraints that still allow for deviation from the constraints, we expect stronger dependencies when tightening the allowed log-ratio deviation from the constraints, which are currently set relatively moderate (rel_tol$_*$ between 0.05 and 0.2, see Table 1).

3.4 Comparing with WLS

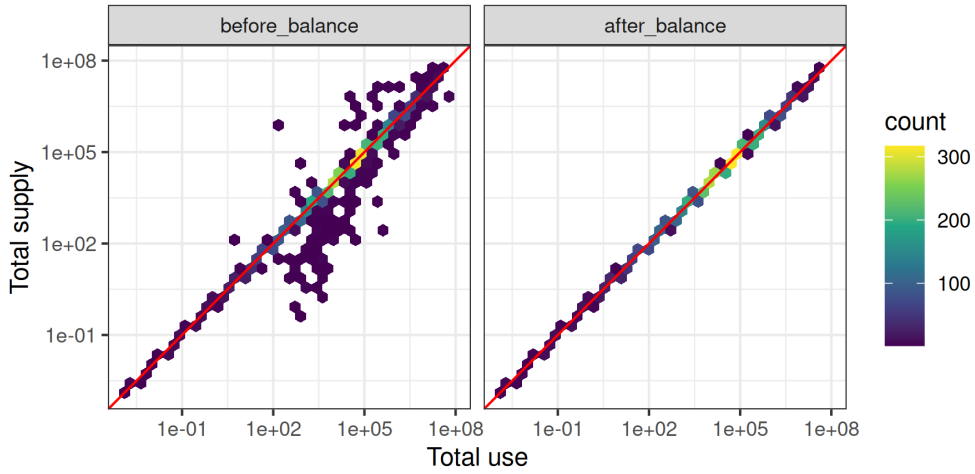
Next, we compare the Bayesian MCMC approach with the deterministic WLS balancing, which is the current default approach for balancing BONSAI (see Section 6.3 for more details). Figure 8 shows a hexbin scatterplot of the log posterior mean from the MCMC model against the log WLS-balanced value for all flows. Most mass lies close to the diagonal, indicating broad consistency between the two methods. Systematic deviations arise from the different ways the two methods resolve imbalances. The WLS scheme optimises multiplicative adjustment factors under hard equality and inequality constraints and a quadratic objective in linear space (Section 6.3). In the simple case of one supply and one use total, this yields a balanced value at the harmonic mean of the two totals, so that strongly imbalanced pairs (e.g. 20 vs 200) are pulled closer to the smaller value. By contrast, the Bayesian model places lognormal priors on flows and enforces balance via soft log-ratio constraints. With roughly symmetric log-scale uncertainties, the posterior tends to concentrate around the geometric mean of the conflicting totals, so the posterior mean often lies closer to the larger value than the WLS solution. Further discrepancies stem from the constraint set itself: the WLS formulation does not include cross-layer consistency constraints (conversion factors), whereas the Bayesian model couples physical and monetary layers explicitly through these additional likelihood terms. The spread around the diagonal in Figure 8 therefore reflects not



(a) Mass layer (tonnes)

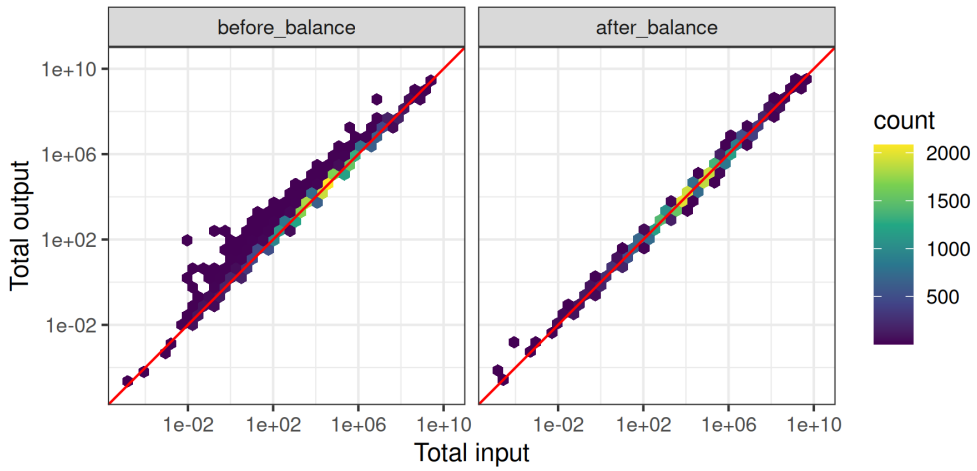


(b) Montary layer (Mill. Euro)

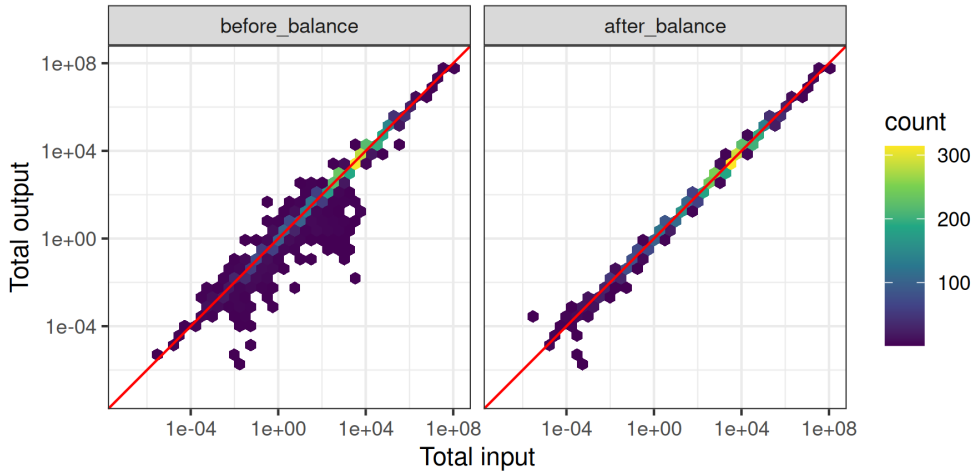


(c) Energy layer (TJ)

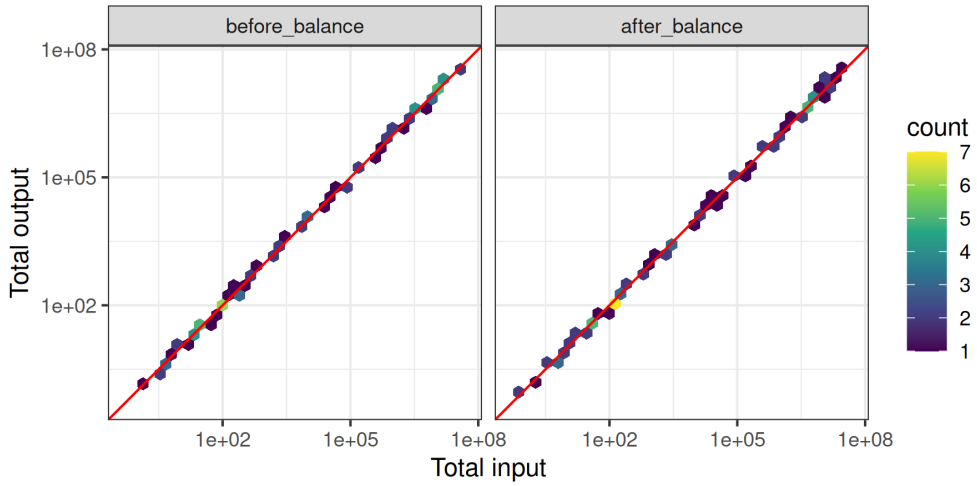
Figure 5: Hexbin scatterplot of the product balance (total supply = total use per product) before and after balancing. The colour scale indicates the number of flows in each bin. The red line marks exact agreement between the two.



(a) Mass layer (tonnes)



(b) Montary layer (Mill. Euro)



(c) Energy layer (TJ)

Figure 6: Hexbin scatterplot of the activity balance (total inputs = total outputs per activity) before and after balancing. The colour scale indicates the number of flows in each bin. The red line marks exact agreement between the two.

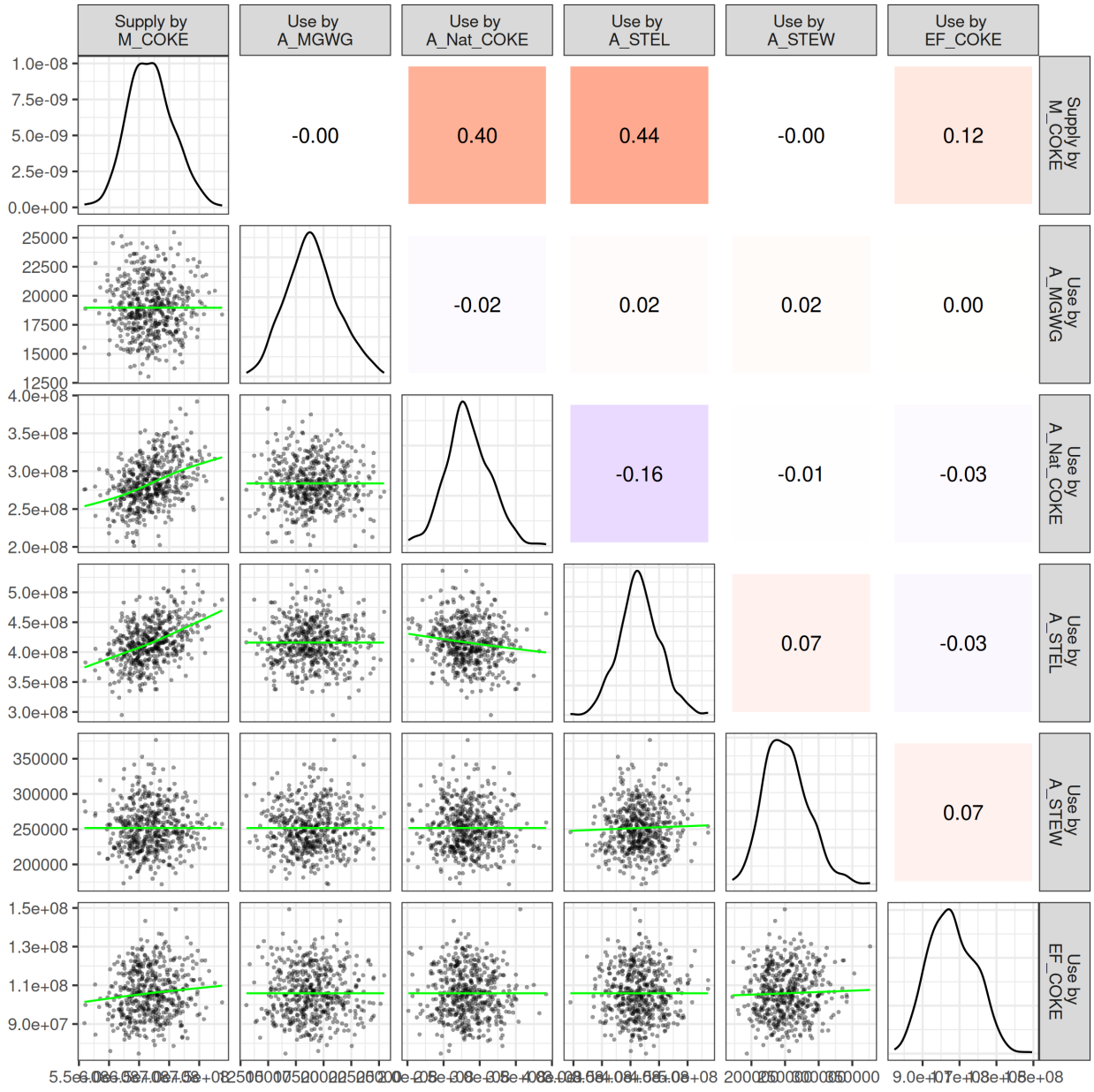


Figure 7: Pairwise posterior correlations between the supply of coke from the Chinese coke market (M_COKE) and its main uses by Chinese activities (A_MGWG, A_Nat_COKE, A_STEL, A_STEW) and the associated combustion flow (EF_COKE). The diagonal panels show marginal posterior densities for each flow. The lower triangle shows posterior draws with fitted regression lines, and the upper triangle reports posterior correlation coefficients, with shading indicating the magnitude of the correlation.

noise, but structural differences in objective function, hard versus soft constraint treatment, and the presence of multi-layer consistency constraints.

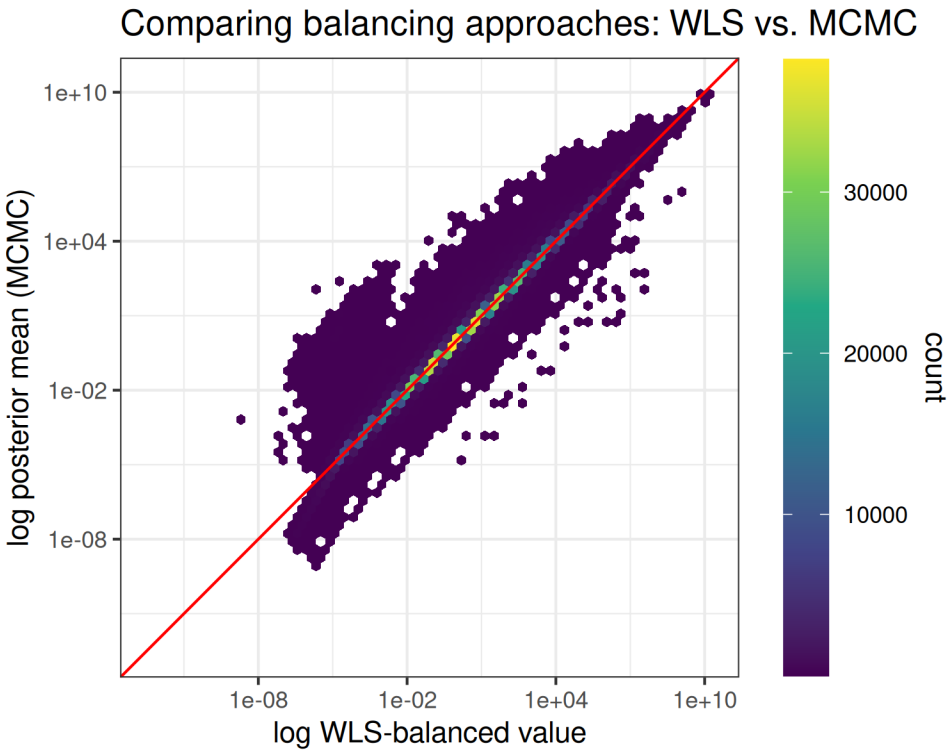


Figure 8: Hexbin scatterplot of the log posterior mean (from the MCMC balancing) against the log WLS-balanced value (Section 6.3). The colour scale indicates the number of flows in each bin. The red line marks exact agreement between the two.

Figure 9 summarises the magnitude of adjustments under the different balancing approaches using empirical cumulative distribution functions (ECDFs) of the absolute relative differences between three pairs of estimates: posterior vs prior (black), WLS vs prior (orange), and posterior vs WLS (blue). For any value on the x-axis, the corresponding y-value gives the fraction of variables whose two compared estimates differ by at most that percentage. The black curve lies consistently above the orange one, indicating that the WLS approach often moves flows further away from the prior than the Bayesian model does. This might be due to the fact that the WLS balancing uses hard constraints, whereas in the Bayesian approach due to soft constraints values are allowed to stay closer the prior.

Figure 10 shows three exemplary examples illustrating how individual flows can be affected very differently by the Bayesian and WLS balancing. Each panel shows one exemplary use flow with the initial BONSAI point estimate (orange dashed line), the corresponding prior density implied by the lognormal prior (orange curve), the WLS-balanced value (purple dashed line), and the posterior sample from the MCMC model (blue histogram). In the top panel (use of CA|M_FORE by CA|A_WOOD), both the posterior and the WLS solution move the flow by more than an order of magnitude away from the initial BONSAI value, and they end up in a similar range, indicating that strong imbalances and constraints force a substantial downward revision of the prior. In the middle panel (use of JP|M_COIL by JP|A_REFN), the posterior distribution is tightly concentrated close to the initial BONSAI estimate, while the WLS solution is pulled almost to zero, illustrating a

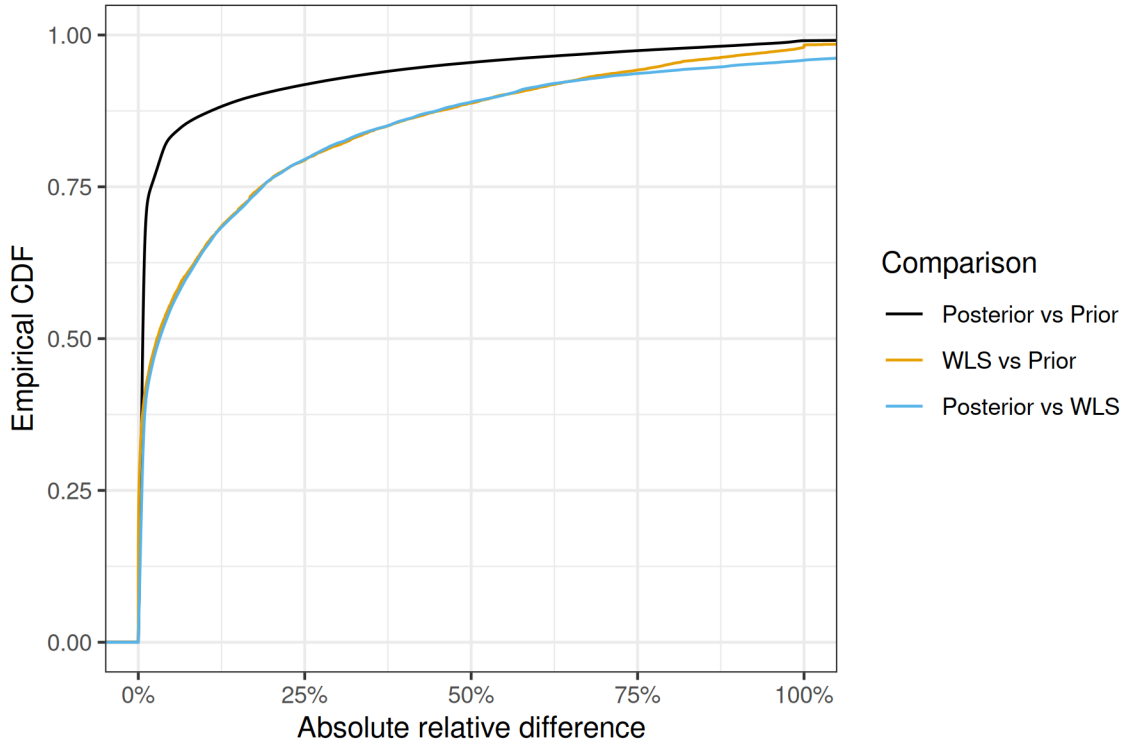


Figure 9: Empirical cumulative distribution functions (ECDF) of the absolute relative differences between the three sets of estimates (Posterior vs Prior, WLS vs Prior, and Posterior vs WLS) across all variables. For any value on the x-axis, the corresponding y-value gives the share of variables for which the two compared estimates differ by at most that percentage, showing that posterior estimates deviate least from the prior, while posterior and WLS estimates display very similar levels of disagreement.

case where the quadratic objective and hard constraints of the WLS formulation produce a much more extreme adjustment than the probabilistic model. The bottom panel (use of CN|M_IRON by CN|A_STEL) shows a more benign case in which prior, posterior, and WLS estimates are all of similar magnitude, with the posterior providing a modest update and an associated uncertainty range around the common central value.

4 Discussion

4.1 Summary of the main findings

The results show that fully Bayesian balancing of a BONSAI-scale hybrid HSUT is computationally feasible and statistically stable. For the configuration reported here, NUTS sampling completed without divergences and 99.98% of all parameters had an $\hat{R} < 1.05$ indicating that independent chains explored the same posterior region and mixed adequately. Most supply and use flows remain close to their initial BONSAI values, with larger adjustments concentrated in those parts of the system where product, activity, and cross-layer constraints expose strong inconsistencies. The Empirical-Bayes inspired prior construction yields prior CVs tightly clustered around 10%, and the posterior largely preserves this scale while redistributing uncertainty up or down depending on how strongly each flow is constrained. The model brings product- and activity-level imbalances in mass, monetary, and energy layers close to equality, and it induces interpretable dependence structures between linked flows, as illustrated by the Chinese coke example.

Compared to the existing WLS procedure, the Bayesian approach is broadly consistent in central tendency but differs systematically in how conflicts are resolved. WLS, with hard equality and inequality constraints and a quadratic objective in linear space, tends to pull strongly imbalanced totals towards a harmonic-mean compromise and can move some flows far from their initial values in order to enforce exact balance. The Bayesian model, in contrast, combines lognormal priors with soft log-ratio constraints and tolerance parameters: under this specification, the posterior flows are closer to geometric means, flows typically stay closer to the initial BONSAI table, and small residual imbalances are tolerated rather than eliminated. This behaviour should be viewed as a modelling trade-off induced by the current choice of priors and constraint tolerances, not as an intrinsic feature: stricter tolerances or more informative priors would push the posterior further away from the prior and closer to exact accounting balance thereby revealing stronger dependency patterns (Section 3.3, at the price of a more challenging posterior geometry and heavier computational burden.

4.2 Limitations and robustness of the current prototype

The current implementation is deliberately pragmatic and should be read as a proof-of-concept rather than a fully specified Bayesian balancing model for BONSAI. Several design choices were made primarily to obtain a workable sampler for a very high-dimensional system. These choices have implications for how the results should be interpreted.

4.2.1 Prior and uncertainty specification

In the present setup, prior uncertainties for supply and use flows ($\sigma_i^{(\text{sup})}$ and $\sigma_i^{(\text{use})}$) are chosen partly for computational robustness. The log-scale standard deviations are calibrated mechanically from observed imbalances between supply and use (Section Section 2.5), with the aim of making the priors and constraints mutually compatible in a system with more than 1.8 Million unknowns. They are intended to be wide enough to make the constraints and priors compatible in such a

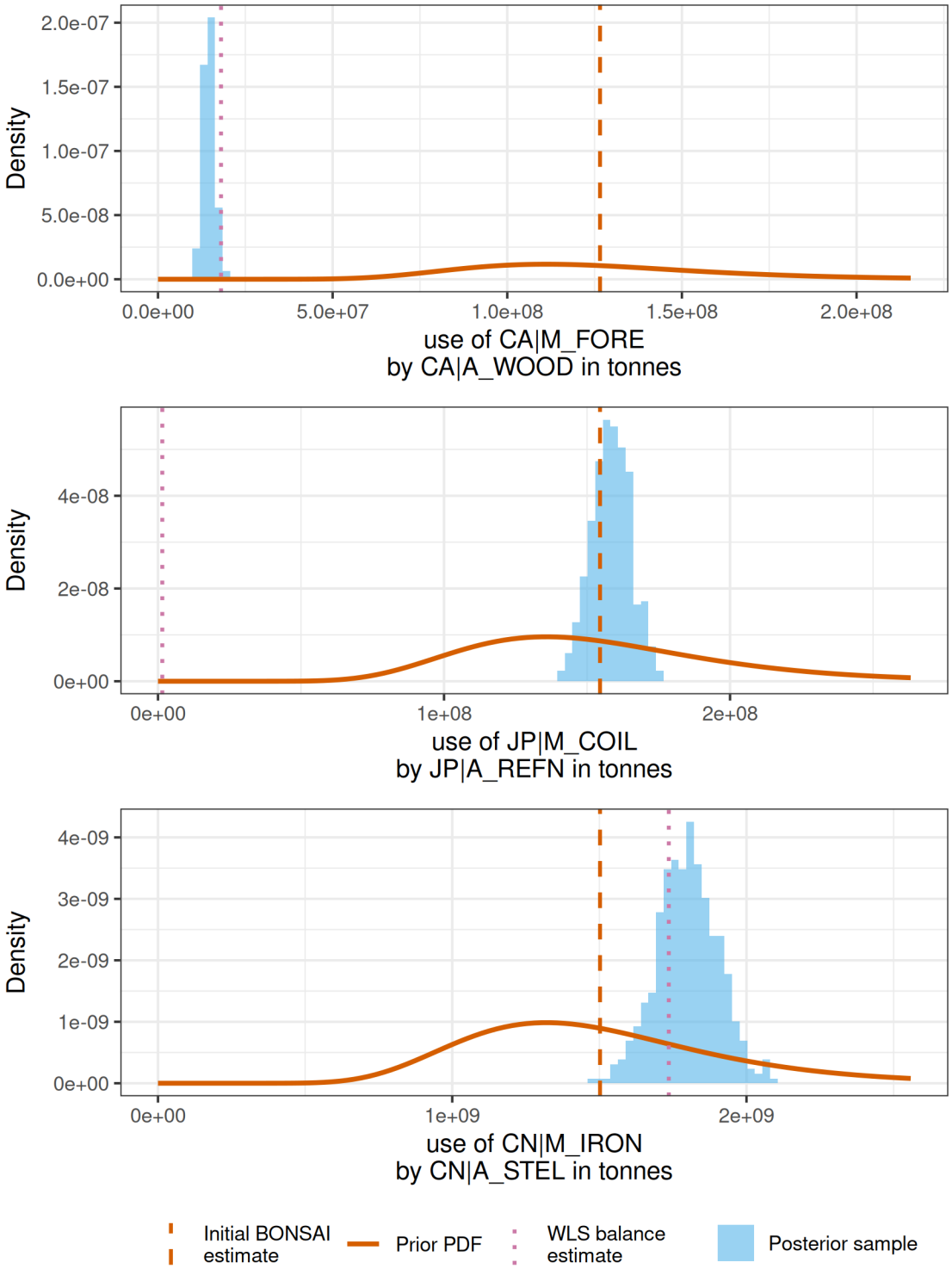


Figure 10: Prior, posterior, and WLS-balanced estimates for three exemplary use flows. The three panels illustrate (top) a case with a large adjustment away from the prior where WLS and the posterior largely agree, (middle) a case where WLS moves the flow far more than the Bayesian model, and (bottom) a case with relatively good agreement between prior, posterior, and WLS estimates.

high-dimensional system, but not too wide to make the posterior untreacable. This calibration does not yet use detailed uncertainty information on data sources or other data quality quality indicators, and should therefore be regarded as a rule-of-thumb procedure rather than a fully informed prior elicitation.

For transfer coefficients and conversion factors, uncertainty is currently represented via global relative uncertainty parameters: a single coefficient of variation for all transfer coefficients, and one per conversion-factor layer. This "one-CV-per-group" specification is convenient and reduces the number of hyperparameters, but it is not fully realistic, as it implicitly treats all coefficients within a group as equally uncertain.

Together, these choices imply that the posterior uncertainty statements are conditional on a relatively simple, globally specified uncertainty structure. They should be interpreted as conditional on this particular calibration of σ and rel_tol used here, rather than as final or unique uncertainty estimates for BONSAI. A more differentiated treatment (e.g. source- or sector-specific σ's, or hierarchical priors for CVs) is an essential direction for subsequent work.

4.2.2 Structural modelling assumptions

Several structural assumptions are embedded in the current prototype:

- **Cell-wise conversion factors:** Each conversion-factor cell (e.g. Euro/t for a given product–region–use combination) is treated as an independent parameter. This is arguably reasonable for prices, which may vary systematically across consumers, but is less realistic for physical conversion factors (e.g. TJ/t), which one might expect to be more homogeneous. An alternative would be to model shared latent conversion factors with multiple noisy observations (partial pooling), especially for physical coefficients.
- **Latent parameters vs. noisy observations:** The model treats supply and use flows as latent parameters with lognormal priors, while transfer coefficients and conversion factors are represented as "true" parameters with Gaussian measurement error around observed BONSAI values. As discussed in Section 2.4.1, these two viewpoints (latent quantity with a prior vs. noisy observation with a likelihood) are closely related mathematically but differ in interpretation and workflow. In practice, the posterior is driven more by the likelihood structure and uncertainty assumptions than by this semantic distinction.
- **Equal weighting of property layers:** All property layers (tonnes, MEuro, items, TJ, service tonnes) enter the product and activity balance constraints with the same statistical weight via a common rel_tol $*$ product and rel_tol $*$ activity. In applications where some layers are known to be more reliable than others, it may be more appropriate to assign property-specific tolerances or weights. This is not yet implemented.
- **Structural zeros:** Flows that are zero in the initial BONSAI HSUT remain zero in the current model. No mechanism is included for creating or deleting flows, so the posterior only re-weights existing non-zero entries. If some structural zeros in BONSAI are in fact unknown small flows, this will not be captured by the present specification.

4.3 Lessons learned on posterior geometry and computation

Experience with the current prototype suggests that the main computational challenges are driven jointly by the sheer number of parameters and by how priors and constraints jointly shape the posterior.

For one-layered small-scale SUTs (100-1000 sectors, only 1 region, only mass layer), a wide range of prior standard deviations and constraint tolerances rel_tol$_*$ led to well-mixed chains. Different settings primarily affected runtime rather than basic convergence. In contrast, for the full BONSAI HSUT (1500 sectors, 49 regions, 4 property layers) the same tuning choices had a much stronger impact on sampler behaviour. For the full BONSAI HSUT, very tight priors and/or small rel_tol$_*$ values tended to over-constrain the system: many flows are simultaneously involved in product balances, activity balances, cross-layer consistency and transfer-coefficient consistency, and strong imbalances in the initial BONSAI table mean that these constraints can pull in incompatible directions.

In such settings, chains frequently failed to move away from their initial values and NUTS adapted to extremely small step sizes, indicating that the sampler struggled to find trajectories consistent with all constraints and priors at once. At the other extreme, making priors and constraint tolerances very wide reduced these problems but led to very long effective runtimes, with slow exploration and occasional mixing issues that are possibly related to a poorly identified posterior.

Two sets of tuning parameters turned out to be particularly influential: the constraint tolerances rel_tol$_*$ and the prior standard deviations σ on the log scale. Increasing rel_tol relaxes the soft constraints and gives the sampler more freedom to adjust flows away from exact balance. Shrinking rel_tol tightens the constraints and makes it harder to reconcile them with the prior when imbalances are large. Similarly, very small σ values leave little room to deviate from the initial BONSAI values, whereas very large σ values effectively hand over most of the influence to the constraints. In practice, the most reliable behaviour for the full HSUT was obtained by choosing σ "wide enough but not excessively wide" (Section Section 2.5) and pairing this with moderate rel_tol$*$ values as reported in Table 1. Under these settings, chains mixed satisfactorily without divergences, while still producing visibly improved balances.

Overall, the computational difficulties encountered so far are not prohibitive, but they highlight where the current combination of priors and constraints creates challenging regimes for MCMC. This, in turn, points to those parts where the initital BONSAI values need revision (due to large prior imbalancess) or revised or prioritised constraints could be beneficial. The present configuration should therefore be seen as a conservative baseline that yields workable inference while leaving room for targeted improvements.

4.4 Future directions and extensions

The following subsections outline possible extensions and refinements of the current prototype. They are not all required for using the present model, but indicate where additional work could substantially improve realism, robustness and computational performance.

4.4.1 Refining prior and uncertainty specification

With the current prototype, posterior uncertainty statements are conditional on the way prior uncertainties have been specified. A natural next step is to replace the current approach by a more systematic uncertainty specification that exploits additional information on data quality for individual flows, or by data source, sector and parameter type.

Firstly, flow- or source-specific uncertainty metadata (e.g. from expert judgment) could be incorporated directly. One simple approach would be to assign each data item (individually or grouped by data source, sector, flow type etc.) a baseline log-scale standard deviation (or coefficient of

variation) and then define $\sigma_{\text{flow}} = \max(\sigma_{\text{base, source}}, \sigma_{\text{from imbalance}})$. High-trust sources would receive a low floor, low-trust sources a higher one, while large observed imbalances could still inflate σ where necessary. Alternatively, the current "equal-weight" imbalance heuristic could be refined by introducing weights in the aggregation step: instead of a simple count k, an effective $k_{\text{eff}} = \sum w_i$ based on source quality could be used, or the supply/use sums entering Δ could themselves be weighted, so that high-quality data tighten the implied prior uncertainty more strongly.

Secondly, the treatment of transfer coefficients and conversion factors could be made more heterogeneous. The current prototype applies global relative uncertainty parameters per parameter type and property layer. The Stan model already supports parameter-specific σ_i. Extending the data preparation to populate these with more differentiated values (for example, by process type, physical vs. monetary factors, or technology group) would allow the priors for these quantities to better reflect domain knowledge.

Finally, more formal Bayesian schemes could be used to learn uncertainty levels from the data. One option would be to place hyperpriors on σ (or on CVs per data source, sector or parameter class) and infer them jointly with the flows, in a hierarchical model. The goal would be to move from a single, manually tuned calibration to a more data-informed and transparent treatment of prior uncertainty.

4.4.2 Refining and extending constraints

The current prototype focuses on internal consistency constraints (product and activity balances, cross-layer conversion consistency, and transfer-coefficient consistency) within the HSUT. While these are essential, they do not yet fully exploit available information that could anchor the system more tightly to observed macro-economic and environmental aggregates.

One avenue for extension is to incorporate additional aggregate constraints. Examples include: (i) GDP-type constraints, where the sum of monetary outputs per country is required to match external national accounts (within some tolerance); (ii) emission constraints, where total GHG emissions per country and/or sector are aligned with reported inventories (e.g. UNFCCC, including residence adjustments); and (iii) technology-specific constraints, such as minimum energy requirements for particular processes. In the here proposed model, these could be introduced as additional (soft) likelihood terms, with their own uncertainty scales reflecting the reliability of the external data and the desired strength of the constraint.

A second direction is to refine the formulation of the existing constraints. For example, the assumption of normality for constraint residuals could be revisited: replacing the normal distribution with a heavier-tailed Student-(t) likelihood would downweight occasional large discrepancies and reduce the influence of individual outliers or mis-specified entries, potentially improving robustness without abandoning the soft-constraint approach.

4.4.3 Computational strategies

Given the size and complexity of the BONSAI HSUT, computational efficiency is a central concern. The current implementation already exploits sparse linear algebra and vectorised likelihood evaluations, but there is scope for further improvement along two lines: faster MCMC for the existing model, and approximate or staged inference schemes.

On the MCMC side, one natural extension is to introduce within-chain parallelisation using Stan's `reduce_sum` functionality. Several of the most expensive components of the log-density – in particular,

the lognormal priors for use flows – can be decomposed into sums over relatively independent subsets. Offloading these partial sums to multiple cores would reduce wall-clock time without changing the underlying posterior. In addition, recent work on Stan compilation and runtime options suggests that substantial speed-ups are sometimes achievable by adjusting compiler flags, math library choices, and threading settings [2].

In parallel, it may be useful to investigate approximate or staged Bayesian methods that are less computationally demanding than full NUTS sampling from the outset. One candidate is Pathfinder [31] or related approximate Bayesian approaches, which can provide a fast approximation to the posterior mode and local curvature. These approximations could either be used directly as a computationally cheaper uncertainty description, or as an initialisation strategy for MCMC: for example, first fit a model with relatively wide constraint tolerances using Pathfinder, and then initialise NUTS from the resulting approximate posterior when running a second stage with tighter tolerances. Such incremental schemes would not remove the need for careful diagnostics, but they might reduce the number of expensive warm-up iterations required to reach the relevant region of the posterior, thereby improving overall computational efficiency.

5 Conclusions and Outlook

This report presents and documents the ambitious, yet successful, development and implementation of a running model of a novel and superior Bayesian balancing approach for SUTs, that with realistic computational requirements handles uncertainty in a consistent manner for one of the largest and most complex SUT-like databases currently available. Due to several pragmatic and simplifying design decisions and the still existing computational challenges, we do not consider the fully probabilistic model proposed here to be a finished model, but rather a starting point for further research that will bring model assumptions closer to reality.

In particular, long runtimes and high RAM usage (for post-processing) make the application of the here proposed Bayesian balancing approach a challenging and demanding task for such a huge and complex system like BONSAI. Yet, most SUTs that currently exist and are used for decision-making (e.g. national monetary SUTs, monetary MRIOs such as EXIOBASE) are much smaller in size and are easier to handle due to containing only one monetary layer. Hence, even without making the adjustments to decrease computational burdens as described above, the here proposed method could already be of great use for national statistical offices (as balancing tool for their national SUTs) or compilation teams of purely monetary MRIOs such as Figaro or EXIOBASE. Moreover, applying the method to a SUT of a smaller, more handleable, size would also ease the implementation and testing/verification of the above discussed adjustments and extensions to the current prototype. However, independent of the size of the SUT, one of the main challenges pertain: assigning prior uncertainties to the flows/factors in a way that does not obstruct computational feasibility.

Acknowledgement

This work was carried out as part of the “Getting the Data Right” project funded by the KR foundation.

Authors contributions

SS: Conceptualization, Methodology, Software, Formal analysis, Investigation, Visualisation, Writing – original draft.

SM, **FY**, **MB**: Data curation, Resources, Validation, Writing – review & editing.

BW: Conceptualization, Supervision, Writing – review & editing.

References

[1] Kadhim Abbood, Gokhan Egilmez, and Ferenc Meszaros. "Multi-Region Input-Output-based Carbon and Energy Footprint Analysis of U.S. Manufacturing". In: *Periodica Polytechnica Social and Management Sciences* 31.2 (June 2023), pp. 91–99. ISSN: 1587-3803. DOI: 10.3311/PPso.19554. (Visited on 07/27/2023).

[2] avehtari. *Options for Improving Stan Sampling Speed*. Aug. 2022. (Visited on 12/22/2025).

[3] Jakub Boratyński. "A Bayesian Approach to Matrix Balancing: Transformation of Industry-Level Data under NACE Revision". In: *Central European Journal of Economic Modelling and Econometrics* 4 (2016), pp. 219–239. ISSN: 2080-0886, 2080-119X. (Visited on 01/23/2024).

[4] Bradley P. Carlin. *Bayes and Empirical Bayes Methods for Data Analysis*. London ; New York : Chapman & Hall, 1996. ISBN: 978-0-412-05611-6. (Visited on 12/09/2025).

[5] Oliver Cencic and Rudolf Frühwirth. "A General Framework for Data Reconciliation—Part I: Linear Constraints". In: *Computers & Chemical Engineering* 75 (Apr. 2015), pp. 196–208. ISSN: 0098-1354. DOI: 10.1016/j.compchemeng.2014.12.004. (Visited on 01/16/2024).

[6] Oliver Cencic and Rudolf Frühwirth. "Data Reconciliation of Nonnormal Observations with Nonlinear Constraints". In: *Journal of Applied Statistics* 45.13 (Oct. 2018), pp. 2411–2428. ISSN: 0266-4763. DOI: 10.1080/02664763.2017.1421916. (Visited on 01/16/2024).

[7] Matteo Giuliani and Andrea Castelletti. "Is Robustness Really Robust? How Different Definitions of Robustness Impact Decision-Making under Climate Change". In: *Climatic Change* 135.3 (Apr. 2016), pp. 409–424. ISSN: 1573-1480. DOI: 10.1007/s10584-015-1586-9. (Visited on 05/08/2024).

[8] Evelyne A. Groen et al. "Methods for Global Sensitivity Analysis in Life Cycle Assessment". In: *The International Journal of Life Cycle Assessment* 22.7 (July 2017), pp. 1125–1137. ISSN: 1614-7502. DOI: 10.1007/s11367-016-1217-3. (Visited on 01/15/2024).

[9] Matthew D. Hoffman and Andrew Gelman. "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." In: *J. Mach. Learn. Res.* 15.1 (2014), pp. 1593–1623. (Visited on 05/13/2025).

[10] Keiichiro Kanemoto et al. "Spatial Variation in Household Consumption-Based Carbon Emission Inventories for 1200 Japanese Cities". In: *Environmental Research Letters* 15.11 (Nov. 2020), p. 114053. ISSN: 1748-9326. DOI: 10.1088/1748-9326/abc045. (Visited on 07/27/2023).

[11] J. Karstensen, G. P. Peters, and R. M. Andrew. "Uncertainty in Temperature Response of Current Consumption-Based Emissions Estimates". In: *Earth System Dynamics* 6.1 (May 2015), pp. 287–309. ISSN: 2190-4979. DOI: 10.5194/esd-6-287-2015. (Visited on 03/24/2021).

[12] Manfred Lenzen, Blanca Gallego, and Richard Wood. "Matrix Balancing Under Conflicting Information". In: *Economic Systems Research* 21.1 (Mar. 2009). Publisher: Routledge _eprint: https://doi.org/10.1080/09535310802688661, pp. 23–44. ISSN: 0953-5314. DOI: 10.1080/09535310802688661. URL: https://doi.org/10.1080/09535310802688661 (visited on 01/03/2023).

[13] Manfred Lenzen, Richard Wood, and Thomas Wiedmann. "Uncertainty Analysis for Multi-Region Input–Output Models – a Case Study of the Uk's Carbon Footprint". In: *Economic Systems Research* 22.1 (Mar. 2010), pp. 43–63. ISSN: 0953-5314. DOI: 10.1080/09535311003661226. (Visited on 08/14/2019).

[14] Manfred Lenzen et al. "Building Eora: A Global Multi-Region Input–Output Database at High Country and Sector Resolution". In: *Economic Systems Research* 25.1 (Mar. 2013), pp. 20–49. ISSN: 0953-5314. DOI: 10.1080/09535314.2013.769938. (Visited on 08/15/2019).

[15] Oleg Lugovoy, Andrey Polbin, and Vladimir Potashnikov. "Bayesian Updating of Input-Output Tables". In: (2015). (Visited on 06/23/2025).

[16] Richard C. Lupton and Julian M. Allwood. "Incremental Material Flow Analysis with Bayesian Inference". In: *Journal of Industrial Ecology* 22.6 (Dec. 2018), pp. 1352–1364. ISSN: 1088-1980. DOI: 10.1111/jiec.12698. (Visited on 06/17/2019).

[17] Stefano Merciai. "An Input-Output Model in a Balanced Multi-Layer Framework". In: *Resources, Conservation and Recycling* 150 (Nov. 2019), p. 104403. ISSN: 0921-3449. DOI: 10.1016/j.resconrec.2019.06.037. (Visited on 08/27/2024).

[18] Daniel Moran et al. "Carbon Footprints of 13000 Cities". In: *Environmental Research Letters* 13.6 (June 2018), p. 064041. ISSN: 1748-9326. DOI: 10.1088/1748-9326/aac72a. (Visited on 07/27/2023).

[19] João F. D. Rodrigues. "A Bayesian Approach to the Balancing of Statistical Economic Data". In: *Entropy* 16.3 (Mar. 2014), pp. 1243–1271. DOI: 10.3390/e16031243. (Visited on 06/17/2019).

[20] Daniel Sanz-Alonso and Omar Al-Ghattas. *A First Course in Monte Carlo Methods*. May 2024. arXiv: 2405.16359 [stat]. (Visited on 10/28/2024).

[21] Simon Schulte, Arthur Jakobs, and Stefan Pauliuk. "Estimating the Uncertainty of the Greenhouse Gas Emission Accounts in Global Multi-Regional Input–Output Analysis". In: *Earth System Science Data* 16.6 (June 2024), pp. 2669–2700. ISSN: 1866-3508. DOI: 10.5194/essd-16-2669-2024. (Visited on 06/30/2024).

[22] Prativa Shrestha and Changyou Sun. "Carbon Emission Flow and Transfer through International Trade of Forest Products". In: *Forest Science* 65.4 (Aug. 2019), pp. 439–451. ISSN: 0015-749X. DOI: 10.1093/forsci/fxz003. (Visited on 07/27/2023).

[23] Konstantin Stadler et al. "EXIOBASE 3: Developing a Time Series of Detailed Environmentally Extended Multi-Regional Input-Output Tables". In: *Journal of Industrial Ecology* 22.3 (2018), pp. 502–515. ISSN: 1530-9290. DOI: 10.1111/jiec.12715. (Visited on 02/21/2019).

[24] Richard Stone, D. G. Champernowne, and J. E. Meade. "The Precision of National Income Estimates". In: *The Review of Economic Studies* 9.2 (1942), p. 111. ISSN: 00346527. DOI: 10.2307/2967664. (Visited on 12/05/2025).

[25] Mike G. Tsionas. "Bayesian Input–Output Table Update Using a Benchmark LASSO Prior". In: *Economic Systems Research* 32.3 (July 2020), pp. 413–427. ISSN: 0953-5314. DOI: 10.1080/09535314.2019.1707170. (Visited on 01/16/2024).

[26] F. Van Der Ploeg. "Reliability and the Adjustment of Sequences of Large Economic Accounting Matrices". In: *Journal of the Royal Statistical Society. Series A (General)* 145.2 (1982), p. 169. ISSN: 00359238. DOI: 10.2307/2981533. JSTOR: 2981533. (Visited on 12/05/2025).

[27] Junyang Wang et al. "Bayesian Material Flow Analysis for Systems with Multiple Levels of Disaggregation and High Dimensional Data". In: *Journal of Industrial Ecology* n/a.n/a (2024). ISSN: 1530-9290. DOI: 10.1111/jiec.13550. (Visited on 10/07/2024).

[28] Harry C. Wilting. "Sensitivity and Uncertainty Analysis in Mrio Modelling; Some Empirical Results with Regard to the Dutch Carbon Footprint". In: *Economic Systems Research* 24.2 (June 2012), pp. 141–171. ISSN: 0953-5314. DOI: 10.1080/09535314.2011.628302. (Visited on 08/02/2023).

[29] Yingfu Xie et al. “Uncertainty and automatic balancing of national accounts with a Swedish application”. en. In: *Statistical Journal of the IAOS* 34.2 (Jan. 2018). Publisher: IOS Press, pp. 263–269. ISSN: 1874-7655. DOI: 10.3233/SJI-170357. URL: https://content.iospress.com/articles/statistical-journal-of-the-iaos/sji170357 (visited on 08/02/2024).

[30] Haoran Zhang et al. “Tracing the Uncertain Chinese Mercury Footprint within the Global Supply Chain Using a Stochastic, Nested Input–Output Model”. In: *Environmental Science & Technology* 53.12 (June 2019), pp. 6814–6823. ISSN: 0013-936X. DOI: 10.1021/acs.est.8b06373. (Visited on 07/27/2023).

[31] Lu Zhang et al. “Pathfinder: Parallel Quasi-Newton Variational Inference”. In: *Journal of Machine Learning Research* 23.306 (2022), pp. 1–49. ISSN: 1533-7928. (Visited on 12/22/2025).

---

6 Appendix

6.1 A detailed literature review on MCMC-based approaches to similar problems

There are few other studies that use MCMC-based approaches to similar problems. Those include (non-extaustively):

1. Cencic and Frühwirth [5] and Cencic and Frühwirth [6] both use MCMC sampling for Data reconciliation when observations follow arbitrary distributions and are subject to linear [5] and non-linear [6] constraints. They model constraints as exact (non-)linear equalitities and use the Independence Metropolis-Hastings sampler [20] to obtain the posterior distribution. They apply their model to a very small system with nine flows and mention the computational challenges for higher dimensions due to the drop of acceptance rate as dimensions grow.
2. Tsionas [25] address the updating of an IO table (or Leontief inverse) when only limited new information is available. They rewrite the IO model as a regression problem, with the entries of the Leontief inverse treated as parameters and the observed productions as dependent variables. They impose a mixture prior on the coefficients of the IO table (normal prior or LASSO prior enforcing sparsity). They apply their method on a 196x196 US IO table. The author explicitly reports severe difficulties with standard MCMC on such a large parameter space due very low acceptance rates and high autocorrelation in the draws.
3. Lugovoy, Polbin, and Potashnikov [15] (conference contribution, not peer-reviewed) develop a Bayesian framework for updating, disaggregating, and balancing IO tables when only partial information is available. The authors model IO coefficients as random variables with prior distributions derived from previously available IO tables (Normal or Beta), and enforce (hard) accounting constraints (row/column sums, aggregation relationships) using an indicator likelihood (1 if constraint is satisfied, 0 otherwise). They use their model on the Russian IO table comprising 69 sectors. The authors discuss computational difficulties and performance limitations due to convergence problems.
4. Boratyński [3] estimate a mapping matrix that converts industry outputs from NACE Rev. 1.1 to NACE Rev. 2 classification. The mapping coefficients are treated as shares forming rows of the mapping matrix, each modelled as a Dirichlet-distributed probability vector. The likelihood enforces accounting consistency between the two industry classifications. Their initial implemntation of the constraint as a zero–one indicator likelihood (i.e. hard constraints), however, poses sampling problems. Thus they switched to modelling the accounting identity as a soft constraint using a Gaussian error model. They apply their approach to a 44×63 mapping matrix.

5. Wang et al. [27] develope a method that can estimate flows and quantify uncertainty in very large and highly disaggregated MFA systems while ensuring reasonable mass balance. They formulate a Bayesian MFA model in which all flows and stock changes are explicit model parameters with priors. Mass-balance constraints are a modelled as soft constraints, so that the posterior does not assign zero probability to small mass imbalances, making sampling feasible in high dimensions. Posterior inference is obtained via the NUTS sampler. They apply their methods to a MFA system with around 60 flows and stocks and discuss computational costs ("90 min to generate 24,000 posterior samples across two chains" Wang et al. [27]).
6. Lupton and Allwood [16] model an MFA system as a deterministic network of mass-conserving processes, expressed through model parameters (external inflows, process efficiencies, and allocation fractions). Prior probability distributions encode initial knowledge about these parameters, and observed data are related to model predictions through likelihood functions. Posterior distributions are obtained using Markov Chain Monte Carlo (PyMC3 / NUTS), producing samples of flows and parameters that quantify uncertainty and update as new information is introduced. They apply their approach to an MFA of the global steel cycle comprising around 80 flows and discuss computational costs ("The case study results took approximately 30 minutes to calculate on a 2 GHz Intel i5 laptop.")

6.2 Additional results

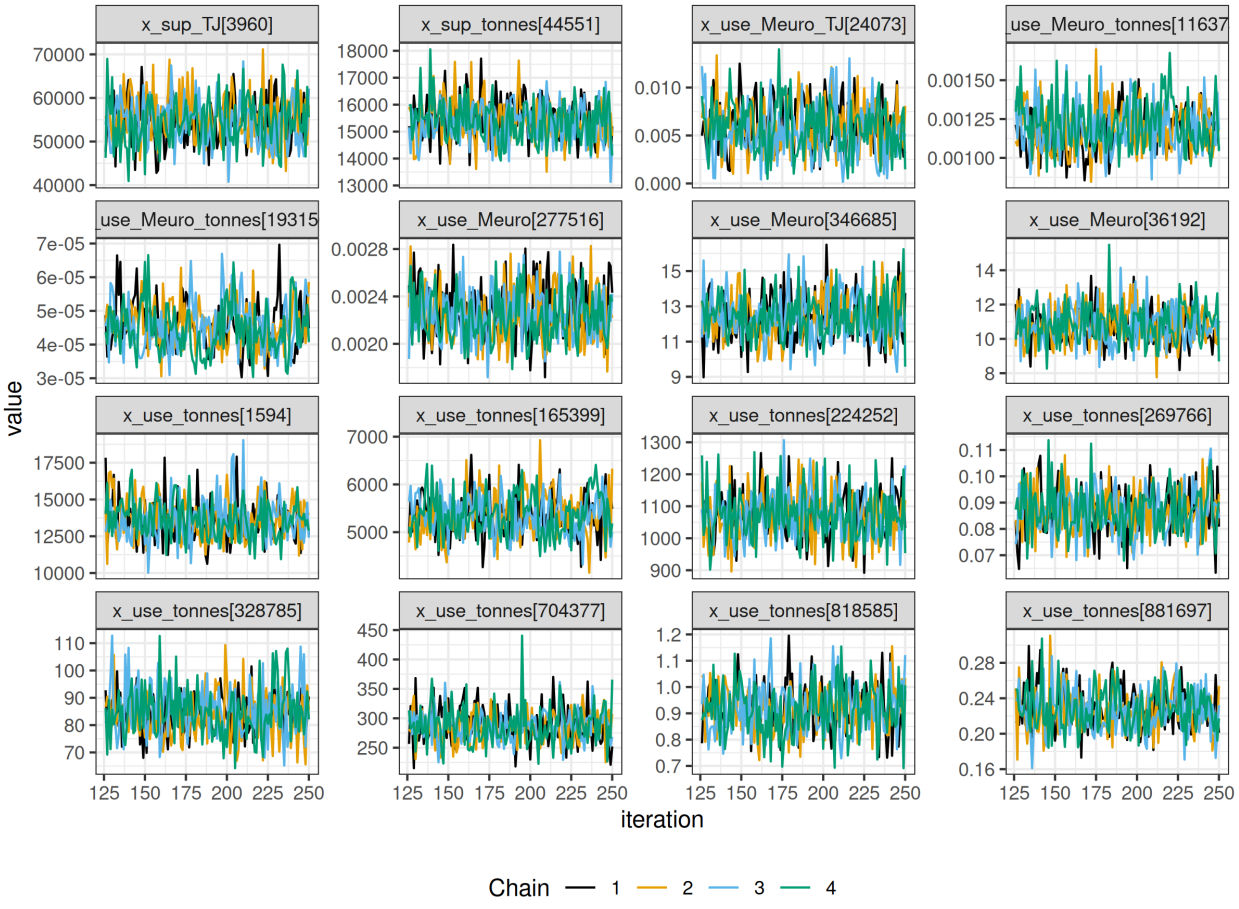


Figure 11: Traceplots for 16 randomly choosen variables.

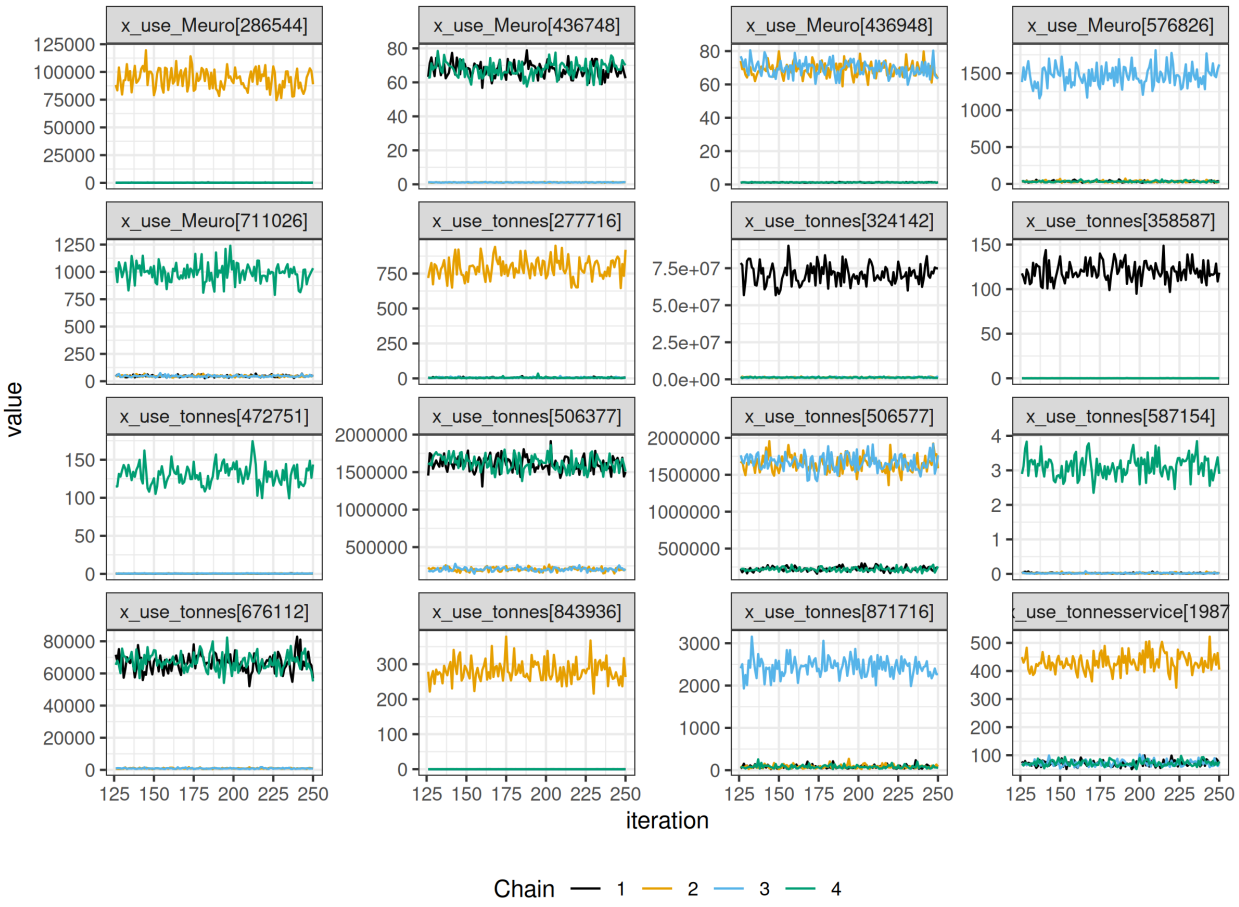


Figure 12: Traceplots for the 16 variables with the highest $\hat{R}$ (worst convergence between chains).

6.3 The WLS balancing approach currently implemented in the BONSAI-IO

In the current version of BONSAI (v2) a Weighted-Least Squares (WLS) approach is used to balance the HSUT across units.

Here is an excerpt of the documentation published on https://hybrid-sut-bonsamurais-bonsai-build-cc0e2d2fdb158dc91033ff98066.gitlab.io/methodology.html#balance.

The balance problem is formulated as follows:

$$\min_{\alpha,\beta} \quad \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} |v_{ijk}|(\alpha_{ijk}-1)^2 + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} |u_{ijk}|(\beta_{ijk}-1)^2$$

$$\begin{aligned}
\text{subject to:} \quad & \sum_{j=1}^{J} v_{ijk}\alpha_{ijk} = \sum_{j=1}^{J} u_{ijk}\beta_{ijk}, && \forall i,\ \forall k, \\
& \sum_{i=1}^{I} v_{ijk}\alpha_{ijk} \le 10\sum_{i=1}^{I} u_{ijk}\beta_{ijk}, && \forall j,\ \forall k, \\
& v_{pjk^*}\alpha_{pjk^*} = r_{b,j}\, v_{bjk^*}\alpha_{bjk^*}, && \forall j, \\
& && \forall p, b \in P_j,\ p \neq b,\ \text{s.t. } r_{b,j} \text{ is defined} \\
& v_{pjk^*}\alpha_{pjk^*} = a_{q,j}\, u_{qjk^*}\,\beta_{qjk^*}, && \forall j, \\
& && \forall p \in P_j,\ \forall q \in P_j,\ p \neq q,\ \text{s.t. } a_{q,j} \text{ is} \\
& \underline{v}_{pjk^*} \le v_{pjk^*}\alpha_{pjk^*} \le \overline{v}_{pjk^*}, && \forall j = 1,\ldots,J,\ \forall p \in P_j,
\end{aligned}$$

where:

- α_{ijk}: Adjustment factor for supply of product i, activity j, and unit k.
- β_{ijk}: Adjustment factor for use of product i, activity j, and unit k.
- v_{ijk}: Initial supply volume of product i by activity j in unit k.
- v_{ibk}: Initial supply volume of the base/determining product b by activity j in unit k.
- u_{ijk}: Initial use volume of product i by activity j in unit k.
- P_j: Set of products co-produced by activity j.
- $r_{b,j}$: By-product ratio indicating by-product b produced per unit of reference product &p& by activity j (v_{bjk}/v_{pjk}).
- $a_{q,j}$: Technical coefficient between reference product
- k^*: The characteristic unit/layer of supply/use.

Constraints

- **Product Balance** ensures total supply of a product equals its total use.
- **Activity Balance** ensures total input to an activity does not exceed its total output.
- **Output Ratio Constraint** ensures the ratio of co-produced products for an activity remains constant.

6.4 Stan programming language

Our balancing model is written in Stan. Stan is a probabilistic programming language for describing probabilisitc models and infering the posterior via MCMC simulations. It turns your model into fast C++ code and uses HMC/NUTS to approximate the posterior. A Stan program is split into blocks, each handling a well-defined stage of the model:

- `data{}`: declare observed inputs with types and dimensions.
- `transformed data{}` (optional): precompute deterministic quantities from the data.
- `parameters{}`: declare unknowns with constraints (e.g. positivity, simplex).
- `transformed parameters{}` (optional): deterministic functions of parameters for readability and efficiency.
- `model{}`: specify priors and likelihood, i.e., the log-density Stan will target.
- `generated quantities{}` (optional): post-processing such as predictions or diagnostics computed after sampling.

The Stan model can be called either from

1. Terminal via `cmdstan`: https://github.com/stan-dev/cmdstan
2. R via `cmdstanR` (used for this work) or `Rstan` (https://mc-stan.org/cmdstanr/ and https://mc-stan.org/rstan/)
3. Python via `cmdstanpy`: https://github.com/stan-dev/cmdstanpy

Links:

- Documentation: https://mc-stan.org/docs/
- Stan user forum: https://discourse.mc-stan.org/

6.5 Mapping between mathematical formulation and Stan implementation

This appendix links each mathematical component of the model to the corresponding implementation in the Stan code in Section 6.6. The goal is not to repeat the full model description, but to provide an easy lookup for readers who want to verify how each equation is executed in code. The symbol-variable mapping is provided in Table Table 2.

Table 2: Mapping of the reported symbols to Stan variable names used in the implementation.

Symbol	Stan variable names
$x_i^{(\text{sup},l)}$	`x_sup_tonnes`, `x_sup_Meuro`, `x_sup_items`, `x_sup_TJ`, `x_sup_tonnesservice`
$x_i^{(\text{use},l)}$	`x_use_tonnes`, `x_use_Meuro`, `x_use_items`, `x_use_TJ`, `x_use_tonnesservice`
$\mu_i^{(\text{sup},l)}$, $\mu_i^{(\text{use},l)}$	`mu_log_sup_tonnes`, `mu_log_sup_Meuro`, `mu_log_sup_items`, `mu_log_sup_TJ`, `mu_log_sup_tonnesservice`, and `mu_log_use_*` equivalents

Symbol	Stan variable names
$\sigma_i^{(\mathrm{sup},l)},\ \sigma_i^{(\mathrm{use},l)}$	`sigma_log_sup_tonnes`, `sigma_log_sup_Meuro`, `sigma_log_sup_items`, `sigma_log_sup_TJ`, `sigma_log_sup_tonnesservice`, and `sigma_log_use_*` equivalents
$x_i^{(cf,l_1\to l_2)}$	`x_sup_Meuro_tonnes`, `x_use_Meuro_tonnes`, `x_sup_tonne_item`, `x_use_tonne_item`, `x_sup_TJ_tonnes`, `x_use_TJ_tonnes`, `x_sup_Meuro_TJ`, `x_use_Meuro_TJ`
$y_i^{(cf,l_1\to l_2)}$	`y_sup_Meuro_tonnes`, `y_use_Meuro_tonnes`, `y_sup_tonne_item`, `y_use_tonne_item`, `y_sup_TJ_tonnes`, `y_use_TJ_tonnes`, `y_sup_Meuro_TJ`, `y_use_Meuro_TJ`
$\sigma_i^{(cf,l_1\to l_2)}$	`sigma_sup_Meuro_tonnes`, `sigma_use_Meuro_tonnes`, `sigma_sup_tonne_item`, `sigma_use_tonne_item`, `sigma_sup_TJ_tonnes`, `sigma_use_TJ_tonnes`, `sigma_sup_Meuro_TJ`, `sigma_use_Meuro_TJ`
$x_i^{(\mathrm{tc})}$	`x_tc`
$\alpha_i^{(\mathrm{tc})},\ \beta_i^{(\mathrm{tc})}$	`alpha_tc`, `beta_tc`
$y_i^{(\mathrm{tc})}$	`y_tc`
$\sigma_i^{(\mathrm{tc})}$	`sigma_tc`
$A_{\mathrm{pr}}^{(l,\mathrm{sup})},\ A_{\mathrm{pr}}^{(l,\mathrm{use})}$	CSR triplets: `w_[unit]_pr_sup`, `v_[unit]_pr_sup`, `l_[unit]_pr_sup`, `w_[unit]_pr_use`, `v_[unit]_pr_use`, `l_[unit]_pr_use`
$A_{\mathrm{ar}}^{(l,\mathrm{sup})},\ A_{\mathrm{ar}}^{(l,\mathrm{use})}$	CSR triplets: `w_[unit]_ar_sup`, `v_[unit]_ar_sup`, `l_[unit]_ar_sup`, `w_[unit]_ar_use`, `v_[unit]_ar_use`, `l_[property layer]_ar_use`
$\mathbf{s}_{\mathrm{pr}}^{(l)},\ \mathbf{u}_{\mathrm{pr}}^{(l)}$	`total_sup_[property layer]_pr`, `total_use_[property layer]_pr`
$\mathbf{o}_{\mathrm{ar}}^{(l)},\ \mathbf{i}_{\mathrm{ar}}^{(l)}$	`total_output_[property layer]_ar`, `total_input_[property layer]_ar`
$\mathrm{rel\_tol}_{\mathrm{product}}$	`rel_tol_product`
$\mathrm{rel\_tol}_{\mathrm{activity}}$	`rel_tol_activity`
ϵ	`epsilon`

6.5.1 Priors for supply and use flows

Mathematical definition:

$$\log x_i^{(\text{sup},l)} \sim \mathcal{N}(\mu_i^{(\text{sup},l)}, \sigma_i^{(\text{sup},l)2}) \qquad \log x_i^{(\text{use},l)} \sim \mathcal{N}(\mu_i^{(\text{use},l)}, \sigma_i^{(\text{use},l)2}) \tag{24}$$

Stan implementation (`model{}`):

```
target += lognormal_lpdf(x_sup_tonnes | mu_log_sup_tonnes, sigma_log_sup_tonnes);
target += lognormal_lpdf(x_use_tonnes | mu_log_use_tonnes, sigma_log_use_tonnes);
// ... repeated equivalently for Meuro, items, TJ, tonnesservice
```

6.5.2 Conversion factors with measurement uncertainty

Mathematical definition:

$$x_i^{(cf,l_1\to l_2)} \sim \mathcal{N}(y_i^{(cf,l_1\to l_2)}, \sigma_i^{(cf,l_1\to l_2)2}) \tag{25}$$

Stan implementation:

```
target += normal_lpdf(x_sup_Meuro_tonnes | y_sup_Meuro_tonnes, sigma_sup_Meuro_tonnes);
target += normal_lpdf(x_use_Meuro_tonnes | y_use_Meuro_tonnes, sigma_use_Meuro_tonnes);
// ... analogous blocks for t/item, TJ/t, €/TJ (supply & use)
```

6.5.3 Transfer coefficients: prior + noisy observation

Mathematical definition:

$$x_i^{(\text{tc})} \sim \text{Beta}(\alpha_i, \beta_i), \qquad y_i^{(\text{tc})} \sim \mathcal{N}(x^{(\text{tc})} * i, \sigma * \text{tc}, i^2) \tag{26}$$

Stan implementation:

```
target += beta_lpdf(x_tc | alpha_tc , beta_tc);
target += normal_lpdf(y_tc | x_tc , sigma_tc);
```

6.5.4 Aggregation to product- and activity–region totals

Mathematical definition:

$$\mathbf{s}^{(l)} * \text{pr} = A^{(l,\text{sup})} * \text{pr}\mathbf{x}^{(\text{sup},l)}, \qquad \mathbf{u}^{(l)} * \text{pr} = A^{(l,\text{use})} * \text{pr}\mathbf{x}^{(\text{use},l)} \tag{27}$$

$$\mathbf{o}^{(l)} * \text{ar} = A^{(l,\text{sup})} * \text{ar}\mathbf{x}^{(\text{sup},l)}, \qquad \mathbf{i}^{(l)} * \text{ar} = A^{(l,\text{use})} * \text{ar}\mathbf{x}^{(\text{use},l)} \tag{28}$$

Stan implementation (example for tonnes; repeated for all units):

```
total_sup_tonnes_pr  = csr_matrix_times_vector(N_pr, N_sup_tonnes, w_tonnes_pr_sup, v_tonnes_pr
total_use_tonnes_pr  = csr_matrix_times_vector(N_pr, N_use_tonnes, w_tonnes_pr_use, v_tonnes_pr
// ... equivalent for activity-region and all other property layer layers
```

6.5.5 Product & activity balance constraints (soft)

Mathematical form:

$$\log \frac{s^{(l)} * \mathrm{pr}, j + \epsilon}{u^{(l)} * \mathrm{pr}, j + \epsilon} \sim \mathcal{N}(0, \mathrm{rel\_tol}^2_{\mathrm{product}}) \tag{29}$$

$$\log \frac{o^{(l)} * \mathrm{ar}, k + \epsilon}{i^{(l)} * \mathrm{ar}, k + \epsilon} \sim \mathcal{N}(0, \mathrm{rel\_tol}^2_{\mathrm{activity}}) \tag{30}$$

Stan implementation (helper function):

```
target += apply_log_ratio_constraint_normal(sup_sub, use_sub, epsilon, rel_tol_product);
target += apply_log_ratio_constraint_normal(output, input, epsilon, rel_tol_activity);
```

Applied separately for all five property layers.

6.5.6 Conversion factor consistency across property layer layers

Mathematical definition:

$$x_{\mathrm{conv},m} = x_m^{(l_1)} \cdot x_m^{(cf, l_1 \to l_2)} \tag{31}$$

$$\log \frac{x_{\mathrm{conv},m} + \epsilon}{x^{(l_2)} * m + \epsilon} \sim \mathcal{N}(0, \mathrm{rel\_tol} * \mathrm{conv_fac}^2) \tag{32}$$

Stan implementation:

```
converted_sup_Meuro_tonnes = x_sup_tonnes[idx_cf_sup_Meuro_tonnes_tonnes] .* x_sup_Meuro_tonnes
reference_sup_Meuro_tonnes = x_sup_Meuro[idx_cf_sup_Meuro_tonnes_Meuro];
target += apply_log_ratio_constraint_normal(converted_sup_Meuro_tonnes,
                                            reference_sup_Meuro_tonnes,
                                            epsilon, rel_tol_conv_fac);
// ... analogous for use side and for t/item, TJ/t, €/TJ
```

6.5.7 Transfer coefficient consistency

Expected supply derived from uses via transfer coefficients:

$$\mathrm{expectedSup} * q = \sum * n :, \mathrm{use}(n) \to \mathrm{sup}(n) = q x^{(\mathrm{tc})} * n \cdot u * \mathrm{par}, \mathrm{use}(n) \tag{33}$$

Constraint:

$$\log \frac{\mathrm{expectedSup} * q + \epsilon}{\mathrm{sup}^{(\mathrm{tonnes})} * q + \epsilon} \sim \mathcal{N}(0, \mathrm{rel\_tol}^2_{\mathrm{tc}}) \tag{34}$$

Stan implementation:

```
for (n in 1:N_tc)
  expected_sup_tc[idx_par_tc_sup[n]] += x_tc[n] * total_use_tonnes_par[idx_par_tc_use[n]];

target += apply_log_ratio_constraint_normal(expected, observed, epsilon, rel_tol_tc);
```

6.5.8 Applying the constraints to subsets of the data

All four constraints (activity & product balance, transfer coefficient and conversion factor constraints) are not applied to *all* supply/use flows but only to a *subset*. In the Stan code those subsets are specified in the `idx_*` vectors.

- The product constraint is applied to commodities (`C_`), market products (`M_`) and energy products (`EF_`). Products being produced but not used or vice versa are excluded. (Stan: `idx_p_constr_[property layer]`).
- The activity constraint is applied to all activities `A_` and markets `M_`, excluding final consumption, markets of fuel mixes, service activities (= activities with physical inputs but only monetary outputs). Activities without inputs/outputs are excluded. (Stan: `idx_a_constr_[property layer]`)
- The conversion factor constraint is applied to those flows which are recorded at least in two units and have a conversion factor. (Stan: `idx_cf_sup_[property layer_numerator]_[property layer_denominator]_[property layer_quantity]`, e.g. `idx_cf_sup_Meuro_tonnes_tonnes` selects the tonne quantities (`x_sup_tonnes`) to multiply by the `Meuro/tonnes` factors (`x_sup_Meuro_tonnes`))
- The transfer cofficient constraint is applied to those flows for which a transfer coefficient is provided. (Stan: `idx_par_sup_tc_constr`)

6.6 The Stan model code

```
functions {
  real apply_log_ratio_constraint_normal(
    vector v1,
    vector v2,
    real   epsilon,
    real   rel_tol) {
      int N = num_elements(v1);
      vector[N] log_ratio = log(v1 + epsilon) - log(v2 + epsilon);
      return normal_lpdf(log_ratio | 0, rel_tol);
    }
}


data {

  // _A) Observation COUNTS
  // Observation COUNTS for flows (SUPPLY / USE) - one per *property layer*
  int<lower=1> N_sup_tonnes;
  int<lower=1> N_sup_Meuro;
```

```
int<lower=1> N_sup_items;
int<lower=1> N_sup_TJ;
int<lower=1> N_sup_tonnesservice;

int<lower=1> N_use_tonnes;
int<lower=1> N_use_Meuro;
int<lower=1> N_use_items;
int<lower=1> N_use_TJ;
int<lower=1> N_use_tonnesservice;


// Observation COUNTS for conversion-factor ("price") layers
int<lower=1> N_sup_Meuro_tonnes;   // € / t            (old)
int<lower=1> N_use_Meuro_tonnes;

int<lower=1> N_sup_tonne_item;     // t / item         (new)
int<lower=1> N_use_tonne_item;

int<lower=1> N_sup_TJ_tonnes;      // TJ / t           (new)
int<lower=1> N_use_TJ_tonnes;

int<lower=1> N_sup_Meuro_TJ;       // € / TJ           (new)
int<lower=1> N_use_Meuro_TJ;

// Number of transfer coefficients
int<lower=1> N_tc;


// _B) Dimension constants---------------------------------------------------------------

int<lower=1> N_p;   // products
int<lower=1> N_a;   // activities
int<lower=1> N_pr;  // product-regions
int<lower=1> N_ar;  // activity-regions
int<lower=1> N_par; // product-activity-regions

// _C) PRIORS: best guess mu & sigma for every flow observation

// __1. SUPPLY
vector[N_sup_tonnes]          mu_log_sup_tonnes;
vector<lower=0>[N_sup_tonnes] sigma_log_sup_tonnes;
vector[N_sup_Meuro]           mu_log_sup_Meuro;
vector<lower=0>[N_sup_Meuro]  sigma_log_sup_Meuro;
vector[N_sup_items]           mu_log_sup_items;
vector<lower=0>[N_sup_items]  sigma_log_sup_items;
vector[N_sup_TJ]              mu_log_sup_TJ;
vector<lower=0>[N_sup_TJ]     sigma_log_sup_TJ;
```

```
vector[N_sup_tonnesservice]   mu_log_sup_tonnesservice;
vector<lower=0>[N_sup_tonnesservice] sigma_log_sup_tonnesservice;

// __2. USE-------------------------------------------------------------------
vector[N_use_tonnes]          mu_log_use_tonnes;
vector<lower=0>[N_use_tonnes] sigma_log_use_tonnes;
vector[N_use_Meuro]           mu_log_use_Meuro;
vector<lower=0>[N_use_Meuro]  sigma_log_use_Meuro;
vector[N_use_items]           mu_log_use_items;
vector<lower=0>[N_use_items]  sigma_log_use_items;
vector[N_use_TJ]              mu_log_use_TJ;
vector<lower=0>[N_use_TJ]     sigma_log_use_TJ;
vector[N_use_tonnesservice]   mu_log_use_tonnesservice;
vector<lower=0>[N_use_tonnesservice] sigma_log_use_tonnesservice;

// __3. Conversion Factors
// OBSERVED prices / conversion factors
// (Normal likelihood centred on y)

// € / t
vector[N_sup_Meuro_tonnes]           y_sup_Meuro_tonnes;
vector<lower=0>[N_sup_Meuro_tonnes]  sigma_sup_Meuro_tonnes;
vector[N_use_Meuro_tonnes]           y_use_Meuro_tonnes;
vector<lower=0>[N_use_Meuro_tonnes]  sigma_use_Meuro_tonnes;

// t / item
vector[N_sup_tonne_item]             y_sup_tonne_item;
vector<lower=0>[N_sup_tonne_item]    sigma_sup_tonne_item;
vector[N_use_tonne_item]             y_use_tonne_item;
vector<lower=0>[N_use_tonne_item]    sigma_use_tonne_item;

// TJ / t
vector[N_sup_TJ_tonnes]              y_sup_TJ_tonnes;
vector<lower=0>[N_sup_TJ_tonnes]     sigma_sup_TJ_tonnes;
vector[N_use_TJ_tonnes]              y_use_TJ_tonnes;
vector<lower=0>[N_use_TJ_tonnes]     sigma_use_TJ_tonnes;

// € / TJ
vector[N_sup_Meuro_TJ]               y_sup_Meuro_TJ;
vector<lower=0>[N_sup_Meuro_TJ]      sigma_sup_Meuro_TJ;
vector[N_use_Meuro_TJ]               y_use_Meuro_TJ;
vector<lower=0>[N_use_Meuro_TJ]      sigma_use_Meuro_TJ;

// __4. Transfer coefficients ------------------------------------------------
vector<lower=0, upper=1>[N_tc] y_tc;                 // Observed transfer coefficients
vector<lower=0>[N_tc] sigma_tc;                      // Measurement errors for y_tc
```

```
// Beta prior parameters for transfer coeffs
vector<lower=0>[N_tc] alpha_tc;
vector<lower=0>[N_tc] beta_tc;


// _D) SPARSE-MATRIX triplets ------------------------------------------------------------
//    (map observations → product/activity regions)
//  identical structure repeated for every *flow* unit
//
// __1. USE ------------------------------------------------------------------
// tonnes
vector[N_use_tonnes] w_tonnes_pr_use;
array[N_use_tonnes]  int v_tonnes_pr_use;
array[N_pr + 1]      int u_tonnes_pr_use;

vector[N_use_tonnes] w_tonnes_ar_use;
array[N_use_tonnes]  int v_tonnes_ar_use;
array[N_ar + 1]      int u_tonnes_ar_use;

// Meuro
vector[N_use_Meuro]  w_Meuro_pr_use;
array[N_use_Meuro]   int v_Meuro_pr_use;
array[N_pr + 1]      int u_Meuro_pr_use;

vector[N_use_Meuro]  w_Meuro_ar_use;
array[N_use_Meuro]   int v_Meuro_ar_use;
array[N_ar + 1]      int u_Meuro_ar_use;

// items
vector[N_use_items]  w_items_pr_use;
array[N_use_items]   int v_items_pr_use;
array[N_pr + 1]      int u_items_pr_use;

vector[N_use_items]  w_items_ar_use;
array[N_use_items]   int v_items_ar_use;
array[N_ar + 1]      int u_items_ar_use;

// TJ
vector[N_use_TJ]     w_TJ_pr_use;
array[N_use_TJ]      int v_TJ_pr_use;
array[N_pr + 1]      int u_TJ_pr_use;

vector[N_use_TJ]     w_TJ_ar_use;
array[N_use_TJ]      int v_TJ_ar_use;
array[N_ar + 1]      int u_TJ_ar_use;

// tonnesservice
```

```
vector[N_use_tonnesservice]  w_tonnesservice_pr_use;
array[N_use_tonnesservice]   int v_tonnesservice_pr_use;
array[N_pr + 1]              int u_tonnesservice_pr_use;

vector[N_use_tonnesservice]  w_tonnesservice_ar_use;
array[N_use_tonnesservice]   int v_tonnesservice_ar_use;
array[N_ar + 1]              int u_tonnesservice_ar_use;

// __2. SUPPLY ------------------------------------------------------------------
// tonnes
vector[N_sup_tonnes] w_tonnes_pr_sup;
array[N_sup_tonnes]  int v_tonnes_pr_sup;
array[N_pr + 1]      int u_tonnes_pr_sup;

vector[N_sup_tonnes] w_tonnes_ar_sup;
array[N_sup_tonnes]  int v_tonnes_ar_sup;
array[N_ar + 1]      int u_tonnes_ar_sup;

// Meuro
vector[N_sup_Meuro]  w_Meuro_pr_sup;
array[N_sup_Meuro]   int v_Meuro_pr_sup;
array[N_pr + 1]      int u_Meuro_pr_sup;

vector[N_sup_Meuro]  w_Meuro_ar_sup;
array[N_sup_Meuro]   int v_Meuro_ar_sup;
array[N_ar + 1]      int u_Meuro_ar_sup;

// items
vector[N_sup_items]  w_items_pr_sup;
array[N_sup_items]   int v_items_pr_sup;
array[N_pr + 1]      int u_items_pr_sup;

vector[N_sup_items]  w_items_ar_sup;
array[N_sup_items]   int v_items_ar_sup;
array[N_ar + 1]      int u_items_ar_sup;

// TJ
vector[N_sup_TJ]     w_TJ_pr_sup;
array[N_sup_TJ]      int v_TJ_pr_sup;
array[N_pr + 1]      int u_TJ_pr_sup;

vector[N_sup_TJ]     w_TJ_ar_sup;
array[N_sup_TJ]      int v_TJ_ar_sup;
array[N_ar + 1]      int u_TJ_ar_sup;

// tonnesservice
vector[N_sup_tonnesservice] w_tonnesservice_pr_sup;
```

```
array[N_sup_tonnesservice]  int v_tonnesservice_pr_sup;
array[N_pr + 1]             int u_tonnesservice_pr_sup;

vector[N_sup_tonnesservice] w_tonnesservice_ar_sup;
array[N_sup_tonnesservice]  int v_tonnesservice_ar_sup;
array[N_ar + 1]             int u_tonnesservice_ar_sup;

// tonnes for transfer coeff constraint
vector[N_sup_tonnes] w_tonnes_par_sup;
array[N_sup_tonnes]  int v_tonnes_par_sup;
array[N_par + 1]      int u_tonnes_par_sup;

vector[N_use_tonnes] w_tonnes_par_use;
array[N_use_tonnes]  int v_tonnes_par_use;
array[N_par + 1]      int u_tonnes_par_use;



// _E) BALANCE-CONSTRAINT index sets ------------------------------------------------------

//   (which product/activities must balance?)
// __1. product balance ------------------------------------------------
int<lower=1> N_p_constr_tonnes;
array[N_p_constr_tonnes]  int<lower=1,upper=N_pr> idx_p_constr_tonnes;
int<lower=1> N_p_constr_Meuro;
array[N_p_constr_Meuro]   int<lower=1,upper=N_pr> idx_p_constr_Meuro;
int<lower=1> N_p_constr_items;
array[N_p_constr_items]   int<lower=1,upper=N_pr> idx_p_constr_items;
int<lower=1> N_p_constr_TJ;
array[N_p_constr_TJ]      int<lower=1,upper=N_pr> idx_p_constr_TJ;
int<lower=1> N_p_constr_tonnesservice;
array[N_p_constr_tonnesservice] int<lower=1,upper=N_pr> idx_p_constr_tonnesservice;

// __2. activity balance -----------------------------------------------
int<lower=1> N_a_constr_tonnes;
array[N_a_constr_tonnes]  int<lower=1,upper=N_ar> idx_a_constr_tonnes;
int<lower=1> N_a_constr_Meuro;
array[N_a_constr_Meuro]   int<lower=1,upper=N_ar> idx_a_constr_Meuro;
int<lower=1> N_a_constr_items;
array[N_a_constr_items]   int<lower=1,upper=N_ar> idx_a_constr_items;
int<lower=1> N_a_constr_TJ;
array[N_a_constr_TJ]      int<lower=1,upper=N_ar> idx_a_constr_TJ;
int<lower=1> N_a_constr_tonnesservice;
array[N_a_constr_tonnesservice] int<lower=1,upper=N_ar> idx_a_constr_tonnesservice;

// __3. conversion-factor layer, INDEX ARRAYS ---------------------------------------------
```

```
  // € / t   (origin)
  array[N_sup_Meuro_tonnes] int<lower=1,upper=N_sup_Meuro>  idx_cf_sup_Meuro_tonnes_Meuro;
  array[N_sup_Meuro_tonnes] int<lower=1,upper=N_sup_tonnes> idx_cf_sup_Meuro_tonnes_tonnes;
  array[N_use_Meuro_tonnes] int<lower=1,upper=N_use_Meuro>  idx_cf_use_Meuro_tonnes_Meuro;
  array[N_use_Meuro_tonnes] int<lower=1,upper=N_use_tonnes> idx_cf_use_Meuro_tonnes_tonnes;

  // t / item
  array[N_sup_tonne_item]   int<lower=1,upper=N_sup_tonnes> idx_cf_sup_tonne_item_tonnes;
  array[N_sup_tonne_item]   int<lower=1,upper=N_sup_items>  idx_cf_sup_tonne_item_items;
  array[N_use_tonne_item]   int<lower=1,upper=N_use_tonnes> idx_cf_use_tonne_item_tonnes;
  array[N_use_tonne_item]   int<lower=1,upper=N_use_items>  idx_cf_use_tonne_item_items;

  // TJ / t
  array[N_sup_TJ_tonnes]    int<lower=1,upper=N_sup_tonnes> idx_cf_sup_TJ_tonnes_tonnes;
  array[N_sup_TJ_tonnes]    int<lower=1,upper=N_sup_TJ>     idx_cf_sup_TJ_tonnes_TJ;
  array[N_use_TJ_tonnes]    int<lower=1,upper=N_use_tonnes> idx_cf_use_TJ_tonnes_tonnes;
  array[N_use_TJ_tonnes]    int<lower=1,upper=N_use_TJ>     idx_cf_use_TJ_tonnes_TJ;

  // € / TJ
  array[N_sup_Meuro_TJ]     int<lower=1,upper=N_sup_TJ>     idx_cf_sup_Meuro_TJ_TJ;
  array[N_sup_Meuro_TJ]     int<lower=1,upper=N_sup_Meuro>  idx_cf_sup_Meuro_TJ_Meuro;
  array[N_use_Meuro_TJ]     int<lower=1,upper=N_use_TJ>     idx_cf_use_Meuro_TJ_TJ;
  array[N_use_Meuro_TJ]     int<lower=1,upper=N_use_Meuro>  idx_cf_use_Meuro_TJ_Meuro;


  // __4. Transfer coeff index arrays ========================================
  array[N_tc] int<lower=1, upper=N_par> idx_par_tc_use;
  array[N_tc] int<lower=1, upper=N_par> idx_par_tc_sup;
  int<lower=1> N_par_sup_tc_constr;
  array[N_par_sup_tc_constr] int<lower=1, upper=N_par> idx_par_sup_tc_constr; // Indices of PAR


  // _F) Tolerances & misc. constants-----------------------------------------------------------

  real<lower=0> rel_tol_product;
  real<lower=0> rel_tol_activity;
  real<lower=0> rel_tol_conv_fac;
  real<lower=0> rel_tol_tc;
  int<lower=0>  grainsize;
  real<lower=1e-12> epsilon;
  real<lower=0> nu;
}


parameters {
  // A) Flows ----------------------------------------------------------------
  vector<lower=0>[N_sup_tonnes]        x_sup_tonnes;
```

```
  vector<lower=0>[N_use_tonnes]        x_use_tonnes;
  vector<lower=0>[N_sup_Meuro]         x_sup_Meuro;
  vector<lower=0>[N_use_Meuro]         x_use_Meuro;
  vector<lower=0>[N_sup_items]         x_sup_items;
  vector<lower=0>[N_use_items]         x_use_items;
  vector<lower=0>[N_sup_TJ]            x_sup_TJ;
  vector<lower=0>[N_use_TJ]            x_use_TJ;
  vector<lower=0>[N_sup_tonnesservice] x_sup_tonnesservice;
  vector<lower=0>[N_use_tonnesservice] x_use_tonnesservice;

  // B) Conversion factors ------------------------------------------------
  // € / t
  vector<lower=0>[N_sup_Meuro_tonnes]  x_sup_Meuro_tonnes;
  vector<lower=0>[N_use_Meuro_tonnes]  x_use_Meuro_tonnes;
  // t / item
  vector<lower=0>[N_sup_tonne_item]    x_sup_tonne_item;
  vector<lower=0>[N_use_tonne_item]    x_use_tonne_item;
  // TJ / t
  vector<lower=0>[N_sup_TJ_tonnes]     x_sup_TJ_tonnes;
  vector<lower=0>[N_use_TJ_tonnes]     x_use_TJ_tonnes;
  // € / TJ
  vector<lower=0>[N_sup_Meuro_TJ]      x_sup_Meuro_TJ;
  vector<lower=0>[N_use_Meuro_TJ]      x_use_Meuro_TJ;

  // C) Transfer coefficient ----------------------------------------------
  vector<lower=0, upper=1>[N_tc] x_tc;             // True transfer coefficients
}



model {
    // _A) Aggregated totals by product-region (pr)  & activity-region (ar) ---------

  //  tonnes
  vector[N_pr] total_sup_tonnes_pr;
  vector[N_pr] total_use_tonnes_pr;
  vector[N_ar] total_output_tonnes_ar;
  vector[N_ar] total_input_tonnes_ar;

  //  Meuro
  vector[N_pr] total_sup_Meuro_pr;
  vector[N_pr] total_use_Meuro_pr;
  vector[N_ar] total_output_Meuro_ar;
  vector[N_ar] total_input_Meuro_ar;

  //  items
  vector[N_pr] total_sup_items_pr;
```

```
vector[N_pr] total_use_items_pr;
vector[N_ar] total_output_items_ar;
vector[N_ar] total_input_items_ar;

//  TJ
vector[N_pr] total_sup_TJ_pr;
vector[N_pr] total_use_TJ_pr;
vector[N_ar] total_output_TJ_ar;
vector[N_ar] total_input_TJ_ar;

//  tonnesservice
vector[N_pr] total_sup_tonnesservice_pr;
vector[N_pr] total_use_tonnesservice_pr;
vector[N_ar] total_output_tonnesservice_ar;
vector[N_ar] total_input_tonnesservice_ar;

// _B) Helper vectors for conversion-factor constraints -----------------------

// € / t  (original)
vector[N_sup_Meuro_tonnes] converted_sup_Meuro_tonnes;
vector[N_use_Meuro_tonnes] converted_use_Meuro_tonnes;
vector[N_sup_Meuro_tonnes] reference_sup_Meuro_tonnes;
vector[N_use_Meuro_tonnes] reference_use_Meuro_tonnes;

// t / item
vector[N_sup_tonne_item]   converted_sup_tonnes_item;
vector[N_use_tonne_item]   converted_use_tonnes_item;
vector[N_sup_tonne_item]   reference_sup_tonnes_item;
vector[N_use_tonne_item]   reference_use_tonnes_item;

// TJ / t
vector[N_sup_TJ_tonnes]    converted_sup_TJ_tonnes;
vector[N_use_TJ_tonnes]    converted_use_TJ_tonnes;
vector[N_sup_TJ_tonnes]    reference_sup_TJ_tonnes;
vector[N_use_TJ_tonnes]    reference_use_TJ_tonnes;

// € / TJ
vector[N_sup_Meuro_TJ]     converted_sup_Meuro_TJ;
vector[N_use_Meuro_TJ]     converted_use_Meuro_TJ;
vector[N_sup_Meuro_TJ]     reference_sup_Meuro_TJ;
vector[N_use_Meuro_TJ]     reference_use_Meuro_TJ;

// _C) Helper vectors for transfer coefficient constraint -------------------
vector[N_par] total_use_tonnes_par;
vector[N_par] total_sup_tonnes_par;
vector[N_par] expected_sup_tc = rep_vector(0.0, N_par);
```

```
// _C) Compute total supply/use input/output  ---------------------------------------------

profile("Compute totals - USE") {

  total_use_tonnes_pr  = csr_matrix_times_vector(N_pr, N_use_tonnes, w_tonnes_pr_use, v_tonne
  total_input_tonnes_ar = csr_matrix_times_vector(N_ar, N_use_tonnes, w_tonnes_ar_use, v_tonn

  total_use_Meuro_pr   = csr_matrix_times_vector(N_pr, N_use_Meuro, w_Meuro_pr_use, v_Meuro_p
  total_input_Meuro_ar = csr_matrix_times_vector(N_ar, N_use_Meuro, w_Meuro_ar_use, v_Meuro_a

  total_use_items_pr   = csr_matrix_times_vector(N_pr, N_use_items, w_items_pr_use, v_items_p
  total_input_items_ar = csr_matrix_times_vector(N_ar, N_use_items, w_items_ar_use, v_items_a

  total_use_TJ_pr      = csr_matrix_times_vector(N_pr, N_use_TJ, w_TJ_pr_use, v_TJ_pr_use, u
  total_input_TJ_ar    = csr_matrix_times_vector(N_ar, N_use_TJ, w_TJ_ar_use, v_TJ_ar_use, u

  total_use_tonnesservice_pr = csr_matrix_times_vector(N_pr, N_use_tonnesservice, w_tonnesser
  total_input_tonnesservice_ar = csr_matrix_times_vector(N_ar, N_use_tonnesservice, w_tonnes
}

profile("Compute totals - SUPPLY") {
  total_sup_tonnes_pr  = csr_matrix_times_vector(N_pr, N_sup_tonnes, w_tonnes_pr_sup, v_tonne
  total_output_tonnes_ar = csr_matrix_times_vector(N_ar, N_sup_tonnes, w_tonnes_ar_sup, v_ton

  total_sup_Meuro_pr   = csr_matrix_times_vector(N_pr, N_sup_Meuro, w_Meuro_pr_sup, v_Meuro_p
  total_output_Meuro_ar = csr_matrix_times_vector(N_ar, N_sup_Meuro, w_Meuro_ar_sup, v_Meuro

  total_sup_items_pr   = csr_matrix_times_vector(N_pr, N_sup_items, w_items_pr_sup, v_items_p
  total_output_items_ar = csr_matrix_times_vector(N_ar, N_sup_items, w_items_ar_sup, v_items

  total_sup_TJ_pr      = csr_matrix_times_vector(N_pr, N_sup_TJ, w_TJ_pr_sup, v_TJ_pr_sup, u
  total_output_TJ_ar   = csr_matrix_times_vector(N_ar, N_sup_TJ, w_TJ_ar_sup, v_TJ_ar_sup, u

  total_sup_tonnesservice_pr   = csr_matrix_times_vector(N_pr, N_sup_tonnesservice, w_tonnes
  total_output_tonnesservice_ar = csr_matrix_times_vector(N_ar, N_sup_tonnesservice, w_tonne
}


// _D) Compute converted layers ------------------------------------------------

profile("Compute converted layers") {
  // naming convention: converted_[[type]]_[[target_unit]]_[[source_unit]]
  // x_[[type]]_[[unit_numerator]]_[[unit_denominator]], e.g. Price input Meuro/tonnes
  // index for conversion factor to convert tonnes input Meuro: idx_cf_[[type]]_Meuro_tonnes_
  // € / t
  converted_sup_Meuro_tonnes = x_sup_tonnes[idx_cf_sup_Meuro_tonnes_tonnes] .* x_sup_Meuro_to
```

```
  converted_use_Meuro_tonnes = x_use_tonnes[idx_cf_use_Meuro_tonnes_tonnes] .* x_use_Meuro_to
  reference_sup_Meuro_tonnes        = x_sup_Meuro[idx_cf_sup_Meuro_tonnes_Meuro];
  reference_use_Meuro_tonnes        = x_use_Meuro[idx_cf_use_Meuro_tonnes_Meuro];

  // t / item
  converted_sup_tonnes_item = x_sup_items[idx_cf_sup_tonne_item_items] .* x_sup_tonne_item;
  converted_use_tonnes_item = x_use_items[idx_cf_use_tonne_item_items] .* x_use_tonne_item;
  reference_sup_tonnes_item  = x_sup_tonnes[idx_cf_sup_tonne_item_tonnes];
  reference_use_tonnes_item  = x_use_tonnes[idx_cf_use_tonne_item_tonnes];

  // TJ / t
  converted_sup_TJ_tonnes = x_sup_tonnes[idx_cf_sup_TJ_tonnes_tonnes] .* x_sup_TJ_tonnes;
  converted_use_TJ_tonnes = x_use_tonnes[idx_cf_use_TJ_tonnes_tonnes] .* x_use_TJ_tonnes;
  reference_sup_TJ_tonnes        = x_sup_TJ[idx_cf_sup_TJ_tonnes_TJ];
  reference_use_TJ_tonnes        = x_use_TJ[idx_cf_use_TJ_tonnes_TJ];

  // € / TJ
  converted_sup_Meuro_TJ = x_sup_TJ[idx_cf_sup_Meuro_TJ_TJ] .* x_sup_Meuro_TJ;
  converted_use_Meuro_TJ = x_use_TJ[idx_cf_use_Meuro_TJ_TJ] .* x_use_Meuro_TJ;
  reference_sup_Meuro_TJ  = x_sup_Meuro[idx_cf_sup_Meuro_TJ_Meuro];
  reference_use_Meuro_TJ  = x_use_Meuro[idx_cf_use_Meuro_TJ_Meuro];
}

// _E) Compute totals required for transfer coeff constraint ---------------------------

profile("Compute transfer coeff totals") {
  // Compute the total supply/use by product-activity-region combination
  // total_sup_tonnes_par is actually equal x_sup_tonnes but just reshaped into a N_par long
  total_sup_tonnes_par  = csr_matrix_times_vector(N_par, N_sup_tonnes, w_tonnes_par_sup, v_to
  total_use_tonnes_par  = csr_matrix_times_vector(N_par, N_use_tonnes, w_tonnes_par_use, v_to

  // Compute expected supply from transfer coeff ---------------------------
  for (n in 1:N_tc) {
    expected_sup_tc[idx_par_tc_sup[n]] += x_tc[n] * total_use_tonnes_par[idx_par_tc_use[n]];
  }

}


// 1) Likelihood for SUPPLY flows (log-normal, vectorised) -------------------
profile("log-likelihood SUPPLY") {
  target += lognormal_lpdf(x_sup_tonnes         | mu_log_sup_tonnes, sigma_log_sup_tonnes);
  target += lognormal_lpdf(x_sup_Meuro          | mu_log_sup_Meuro, sigma_log_sup_Meuro);
  target += lognormal_lpdf(x_sup_items          | mu_log_sup_items, sigma_log_sup_items);
  target += lognormal_lpdf(x_sup_TJ             | mu_log_sup_TJ, sigma_log_sup_TJ);
  target += lognormal_lpdf(x_sup_tonnesservice  | mu_log_sup_tonnesservice, sigma_log_sup_ton
}
```

```
// 2) Likelihood for USE flows ------------------------------------------------
profile("log-likelihood USE") {
  target += lognormal_lpdf(x_use_tonnes         | mu_log_use_tonnes, sigma_log_use_tonnes);
  target += lognormal_lpdf(x_use_Meuro          | mu_log_use_Meuro, sigma_log_use_Meuro);
  target += lognormal_lpdf(x_use_items          | mu_log_use_items, sigma_log_use_items);
  target += lognormal_lpdf(x_use_TJ             | mu_log_use_TJ, sigma_log_use_TJ);
  target += lognormal_lpdf(x_use_tonnesservice  | mu_log_use_tonnesservice, sigma_log_use_to
}

// 3) Price / conversion-factor likelihoods  (Normal) -------------------------
profile("log-likelihood Conversion Factors") {

  // € / t
  target += normal_lpdf(x_sup_Meuro_tonnes | y_sup_Meuro_tonnes, sigma_sup_Meuro_tonnes);
  target += normal_lpdf(x_use_Meuro_tonnes | y_use_Meuro_tonnes, sigma_use_Meuro_tonnes);
  // t / item
  target += normal_lpdf(x_sup_tonne_item   | y_sup_tonne_item, sigma_sup_tonne_item);
  target += normal_lpdf(x_use_tonne_item   | y_use_tonne_item, sigma_use_tonne_item);
  // TJ / t
  target += normal_lpdf(x_sup_TJ_tonnes    | y_sup_TJ_tonnes, sigma_sup_TJ_tonnes);
  target += normal_lpdf(x_use_TJ_tonnes    | y_use_TJ_tonnes, sigma_use_TJ_tonnes);
  // € / TJ
  target += normal_lpdf(x_sup_Meuro_TJ     | y_sup_Meuro_TJ, sigma_sup_Meuro_TJ);
  target += normal_lpdf(x_use_Meuro_TJ     | y_use_Meuro_TJ, sigma_use_Meuro_TJ);

}

profile("Prior for transfer coefficients") {
  target += beta_lpdf(  x_tc | alpha_tc , beta_tc );   // x_tc ~ beta(...)
  target += normal_lpdf( y_tc | x_tc     , sigma_tc ); // y_tc ~ normal(...)
}



// 4) PRODUCT-balance constraints (all five flow units) ----------------------
profile("Product balance constraint") {

  {
    vector[N_p_constr_tonnes] sup_sub = total_sup_tonnes_pr[idx_p_constr_tonnes];
    vector[N_p_constr_tonnes] use_sub = total_use_tonnes_pr[idx_p_constr_tonnes];
    target += apply_log_ratio_constraint_normal(sup_sub, use_sub, epsilon, rel_tol_product);
  }
  {
    vector[N_p_constr_Meuro] sup_sub = total_sup_Meuro_pr[idx_p_constr_Meuro];
    vector[N_p_constr_Meuro] use_sub = total_use_Meuro_pr[idx_p_constr_Meuro];
    target += apply_log_ratio_constraint_normal(sup_sub, use_sub, epsilon, rel_tol_product);
  }
```

```
  {
    vector[N_p_constr_items] sup_sub = total_sup_items_pr[idx_p_constr_items];
    vector[N_p_constr_items] use_sub = total_use_items_pr[idx_p_constr_items];
    target += apply_log_ratio_constraint_normal(sup_sub, use_sub, epsilon, rel_tol_product);
  }
  {
    vector[N_p_constr_TJ] sup_sub = total_sup_TJ_pr[idx_p_constr_TJ];
    vector[N_p_constr_TJ] use_sub = total_use_TJ_pr[idx_p_constr_TJ];
    target += apply_log_ratio_constraint_normal(sup_sub, use_sub, epsilon,
    rel_tol_product);
  }
  {
    vector[N_p_constr_tonnesservice] sup_sub = total_sup_tonnesservice_pr[idx_p_constr_tonnes
    vector[N_p_constr_tonnesservice] use_sub = total_use_tonnesservice_pr[idx_p_constr_tonnes
    target += apply_log_ratio_constraint_normal(sup_sub, use_sub, epsilon, rel_tol_product);
  }
}
// 5) ACTIVITY-balance constraints ----------------------------------------
profile("Activity balance constraint") {

  {
    vector[N_a_constr_tonnes] output = total_output_tonnes_ar[idx_a_constr_tonnes];
    vector[N_a_constr_tonnes] input  = total_input_tonnes_ar[idx_a_constr_tonnes];
    target += apply_log_ratio_constraint_normal(output, input, epsilon, rel_tol_activity);
  }
  {
    vector[N_a_constr_Meuro] output = total_output_Meuro_ar[idx_a_constr_Meuro];
    vector[N_a_constr_Meuro] input  = total_input_Meuro_ar[idx_a_constr_Meuro];
    target += apply_log_ratio_constraint_normal(output, input, epsilon, rel_tol_activity);
  }
  {
    vector[N_a_constr_items] output = total_output_items_ar[idx_a_constr_items];
    vector[N_a_constr_items] input  = total_input_items_ar[idx_a_constr_items];
    target += apply_log_ratio_constraint_normal(output, input, epsilon, rel_tol_activity);
  }
  {
    vector[N_a_constr_TJ] output = total_output_TJ_ar[idx_a_constr_TJ];
    vector[N_a_constr_TJ] input  = total_input_TJ_ar[idx_a_constr_TJ];
    target += apply_log_ratio_constraint_normal(output, input, epsilon, rel_tol_activity);
  }
  {
    vector[N_a_constr_tonnesservice] output =  total_output_tonnesservice_ar[idx_a_constr_to
    vector[N_a_constr_tonnesservice] input  =  total_input_tonnesservice_ar[idx_a_constr_ton
    target += apply_log_ratio_constraint_normal(output, input, epsilon,   rel_tol_activity);
  }
}
// 6) Conversion-factor constraints   (all four layers) ------------------------
```

```
  profile("Conversion factor constraint") {


    // € / t
    {
      target += apply_log_ratio_constraint_normal(converted_sup_Meuro_tonnes, reference_sup_Meu
      target += apply_log_ratio_constraint_normal(converted_use_Meuro_tonnes, reference_use_Meu
    }
    // t / item
    {
      target += apply_log_ratio_constraint_normal(converted_sup_tonnes_item, reference_sup_tonn
      target += apply_log_ratio_constraint_normal(converted_use_tonnes_item,reference_use_tonne
    }
    // TJ / t
    {
      target += apply_log_ratio_constraint_normal(converted_sup_TJ_tonnes, reference_sup_TJ_ton
      target += apply_log_ratio_constraint_normal(converted_use_TJ_tonnes, reference_use_TJ_ton
    }
    // € / TJ
    {
      target += apply_log_ratio_constraint_normal(converted_sup_Meuro_TJ, reference_sup_Meuro_T
      target += apply_log_ratio_constraint_normal(converted_use_Meuro_TJ, reference_use_Meuro_T
    }
  }

  // 7) Transfer coefficient constraints -------------------------------------
  profile("Transfer coefficient constraint") {
    vector[N_par_sup_tc_constr] expected = expected_sup_tc[idx_par_sup_tc_constr];
    vector[N_par_sup_tc_constr] observed = total_sup_tonnes_par[idx_par_sup_tc_constr];
    target += apply_log_ratio_constraint_normal(expected, observed, epsilon, rel_tol_tc);
  }

}
```