How to use DeCovarT: a toy example with two genes and two cell populations

true true true

1 Model

We introduce the following notations:

- $(\boldsymbol{y} = (y_{gi}) \in \mathbb{R}_+^{G\times N}$ is the global bulk transcriptomic expression, measured in N individuals.
- $\boldsymbol{X} = (x_{gj}) \in \mathcal{M}_{\mathbb{R}^{G\times J}}$ the signature matrix of the mean expression of G genes in J purified cell populations.
- $\boldsymbol{p} = (p_{ji}) \in]0,1[^{J\times N}$ the unknown relative proportions of cell populations in N samples

As in most traditional deconvolution models, we assume that the total bulk expression can be reconstructed by summing the individual contributions of each cell population weighted by its frequency, as stated explicitly in the following linear matricial relationship (Eq.(1)):

$$\boldsymbol{y} = \boldsymbol{X} \times \boldsymbol{p} \tag{1}$$

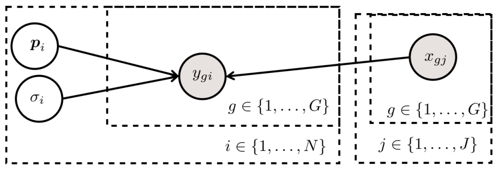
In addition, we consider the following unit simplex constraint on the cellular ratios (Eq.(2)):

$$\begin{cases} \sum_{j=1}^{J} p_j = 1 \\ \forall j \in \widetilde{J} \quad p_j \geq 0 \end{cases} \tag{2}$$

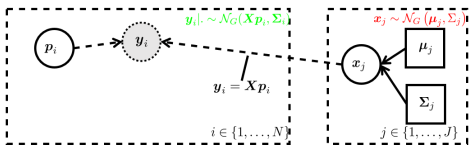
1.1 Rationale of the new generative model

However, in real conditions with technical and environmental variability, the strict linearity of the deconvolution does not strictly hold. Thus, an additional error term is usually added, assumed to follow a *homoscedastic* zero-centred Gaussian distribution and with pairwise independent response measures while the exogenous variables (here, the purified expression profiles) are supposed determined: this set of conditions is referred to as the Gaussian-Markow assumptions. In that configuration, the MLE (maximum likelihood estimate) that bast describes this standard linear model is equal to the ordinary least squares (OLS) estimate.

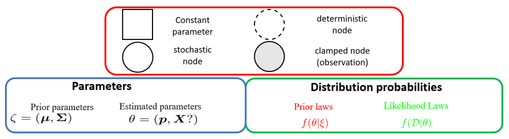
In contrast to this canonical approach, in DeCovarT, we relax the *exogeneity* property by treating exogenous variables $\boldsymbol{X}$ as random variables rather than determined measures, in a process close to the approach of the DSection algorithm [1]. However, to our knowledge, we are the first to weaken the independence assumption between observations by explicitly incorporating the intrinsic covariance structure of the transcriptome of each purified cell population. To do so, we conjecture that the G-dimensional vector $\boldsymbol{x}_j$ characterising the transcriptomic expression of each cell population follows a multivariate Gaussian distribution: $\boldsymbol{x}_j \sim \mathcal{N}_G(\boldsymbol{\mu}_{.j}, \boldsymbol{\Sigma}_j)$, with $\boldsymbol{\mu}_{.j}$ the mean purified transcriptomic expression and $\boldsymbol{\Sigma}_j$ the covariance matrix, that we constrain to be positive-definite and of full rank and that is inferred using the output of the gLasso algorithm [2]. We display respectively the graphical models associated to the standard linear deconvolution model and our new innovative generative model used by the DeCovarT algorithm in subfigures a) and b), in Fig.1.



(a) Visual representation of the linear regression graphical model


$\boldsymbol{y}_i|. \sim \mathcal{N}_G(\boldsymbol{X}\boldsymbol{p}_i, \boldsymbol{\Sigma}_i)$
$\boldsymbol{x}_j \sim \mathcal{N}_G(\boldsymbol{\mu}_j, \Sigma_j)$
$\boldsymbol{p}_i$
$\boldsymbol{y}_i$
$\boldsymbol{y}_i = \boldsymbol{X}\boldsymbol{p}_i$
$\boldsymbol{x}_j$
$\boldsymbol{\mu}_j$
$\boldsymbol{\Sigma}_j$
$i \in \{1,\dots,N\}$
$j \in \{1,\dots,J\}$

(b) Visual representation of the graphical model underlying the DeCovarT generative model


Constant parameter
deterministic node
stochastic node
clamped node (observation)
Parameters
Prior parameters $\zeta = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$
Estimated parameters $\theta = (\boldsymbol{p}, \boldsymbol{X}?)$
Distribution probabilities
Prior laws $f(\theta|\xi)$
Likelihood Laws $f(\mathcal{D}|\theta)$

(c) Legend displaying the main symbols and laws used in a graphical model.

Figure 1: We use the standard graphical convention of graphical models, as depicted in RevBayes webpage. For identifiability reasons, we conjecture that all variability arises from the stochastic nature of the covariates.

1.2 Derivation of the log-likelihood

First, we *plugged-in* the mean and covariance parameters $\zeta_j = \left(\boldsymbol{\mu}_{.j}, \boldsymbol{\Sigma}_j\right)$ inferred in the previous step. Then, by letting $\boldsymbol{\zeta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\quad \boldsymbol{\mu} = (\boldsymbol{\mu}_{.j})_{j \in \widetilde{J}} \in \mathcal{M}_{G \times J}$, $\quad \boldsymbol{\Sigma} \in \mathcal{M}_{G \times G}$ the known parameters and $\boldsymbol{p}$ the unknown cellular ratios, the conditional distribution $\boldsymbol{y}|(\boldsymbol{\zeta}, \boldsymbol{p})$ is the convolution of pairwise independent multivariate Gaussian distributions, which is also a multivariate Gaussian distribution (Eq.(3)), deduced from the *affine invariant* property of Gaussian distributions.

$$\boldsymbol{y}|(\boldsymbol{\zeta}, \boldsymbol{p}) \sim \mathcal{N}_G(\boldsymbol{\mu}\boldsymbol{p}, \boldsymbol{\Sigma}) \text{ with } \boldsymbol{\mu} = (\boldsymbol{\mu}_{.j})_{j \in \widetilde{J}}, \quad \boldsymbol{p} = (p_1, \dots, p_J) \text{ and } \boldsymbol{\Sigma} = \sum_{j=1}^{J} p_j^2 \boldsymbol{\Sigma}_j \tag{3}$$

From Eq.(3), we readily compute the associated conditional log-likelihood (Eq.(4)):

$$\ell_{\boldsymbol{y}|\boldsymbol{\zeta}}(\boldsymbol{p}) = C + \log\left(\operatorname{Det}\left(\sum_{j=1}^{J} p_j^2 \boldsymbol{\Sigma}_j\right)^{-1}\right) - \frac{1}{2}(\boldsymbol{y} - \boldsymbol{p}\boldsymbol{\mu})^\top \left(\sum_{j=1}^{J} p_j^2 \boldsymbol{\Sigma}_j\right)^{-1} (\boldsymbol{y} - \boldsymbol{p}\boldsymbol{\mu}) \tag{4}$$

1.3 First and second-order derivation of the unconstrained DeCovarT log-likelihood function

The stationary points of a function and notably maxima, are given by the roots (the values at which the function crosses the x-axis) of its gradient, in our context, the vector: $\nabla \ell : \mathbb{R}^J \to \mathbb{R}^J$ evaluated at point $\nabla \ell(\boldsymbol{p}) :]0,1[^J \to \mathbb{R}^J$. Since the computation is the same for any cell ratio p_j, we give an explicit formula for only one of them (Eq.(5)):

$$\begin{aligned}
\frac{\partial \ell_{\boldsymbol{y}|\boldsymbol{\zeta}}(\boldsymbol{p})}{\partial p_j} &= \tfrac{\partial \log(\operatorname{Det}(\boldsymbol{\Theta}))}{\partial p_j} - \tfrac{1}{2}\left[\tfrac{\partial (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})^\top}{\partial p_j} \boldsymbol{\Theta}(\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p}) + (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})^\top \tfrac{\partial \boldsymbol{\Theta}}{\partial p_j}(\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p}) + (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta} \tfrac{\partial (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})}{\partial p_j}\right] \\
&= -\operatorname{Tr}\left(\boldsymbol{\Theta} \tfrac{\partial \boldsymbol{\Sigma}}{\partial p_j}\right) - \tfrac{1}{2}\left[-\boldsymbol{\mu}_{.j}^\top \boldsymbol{\Theta}(\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p}) - (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})^\top \Theta \tfrac{\partial \Sigma}{\partial p_j} \Theta (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p}) - (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta}\boldsymbol{\mu}_{.j}\right] \\
&= -2p_j \operatorname{Tr}(\boldsymbol{\Theta}\boldsymbol{\Sigma}_j) + (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta}\boldsymbol{\mu}_{.j} + p_j (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta}\Sigma_j \boldsymbol{\Theta}(\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})
\end{aligned} \tag{5}$$

Since the solution to $\nabla\left(\ell_{\boldsymbol{y}|\boldsymbol{\zeta}}(\boldsymbol{p})\right) = 0$ is not closed, we had to approximate the MLE using iterated numerical optimisation methods. Some of them, such as the Levenberg–Marquardt algorithm, require a second-order approximation of the function, which needs the computation of the Hessian matrix. Deriving once more Eq.(6) yields the Hessian matrix, $\mathbf{H} \in \mathcal{M}_{J \times J}$ is given by:

$$
\begin{aligned}
\mathbf{H}_{i,i} = \frac{\partial^2 \ell}{\partial^2 p_i} &= -2 \operatorname{Tr}\left(\boldsymbol{\Theta}\boldsymbol{\Sigma}_i\right) + 4p_i^2 \operatorname{Tr}\left(\left(\boldsymbol{\Theta}\boldsymbol{\Sigma}_i\right)^2\right) - 2p_i(\boldsymbol{y} - \boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta}\boldsymbol{\Sigma}_i\boldsymbol{\Theta}\boldsymbol{\mu}_{\boldsymbol{.i}} - \boldsymbol{\mu}_{.i}^\top \boldsymbol{\Theta}\boldsymbol{\mu}_{\boldsymbol{.i}} - \\
& 2p_i(\boldsymbol{y} - \boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta}\boldsymbol{\Sigma}_i\boldsymbol{\Theta}\boldsymbol{\mu}_{.i\, -(\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta}\left(4p_i^2 \boldsymbol{\Sigma}_i\boldsymbol{\Theta}\boldsymbol{\Sigma}_i - \boldsymbol{\Sigma}_i\right)\boldsymbol{\Theta}(\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})}, \quad i \in \widetilde{J} \\
\mathbf{H}_{i,j} = \frac{\partial^2 \ell}{\partial p_i \partial p_j} &= 4p_j p_i \operatorname{Tr}\left(\boldsymbol{\Theta}\boldsymbol{\Sigma}_j\boldsymbol{\Theta}\boldsymbol{\Sigma}_i\right) - 2p_i(\boldsymbol{y} - \boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta}\boldsymbol{\Sigma}_i\boldsymbol{\Theta}\boldsymbol{\mu}_{\boldsymbol{.j}} - \boldsymbol{\mu}_{.i}^\top \boldsymbol{\Theta}\boldsymbol{\mu}_{\boldsymbol{.j}} - \\
& 2p_j(\boldsymbol{y} - \boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta}\boldsymbol{\Sigma}_j\boldsymbol{\Theta}\boldsymbol{\mu}_{.i\, -4p_i p_j (\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})^\top \boldsymbol{\Theta}\boldsymbol{\Sigma}_i\boldsymbol{\Theta}\boldsymbol{\Sigma}_j\boldsymbol{\Theta}(\boldsymbol{y}-\boldsymbol{\mu}\boldsymbol{p})}, \quad (i,j) \in \widetilde{J}^2, i \neq j
\end{aligned}
\tag{6}
$$

in which the coloured sections pair one by one with the corresponding coloured sections of the gradient, given in Eq.(5). Matrix calculus can largely ease the derivation of complex algebraic expressions, thus we remind in Appendix Matrix calculus relevant matrix properties and derivations [1].

However, the explicit formulas for the gradient and the Hessian matrix of the log-likelihood function, given in Eq.(5) and Eq.(6) respectively, do not take into account the simplex constraint assigned to the ratios. While some optimisation methods use heuristic methods to solve this problem, we consider alternatively a reparametrised version of the problem, detailed comprehensively in Appendix Reparametrised log-likelihood.

1.4 Iterated optimisation

The MLE is traditionally retrieved from the roots of the gradient of the log-likelihood. However, in our generative framework, cancelling the gradient of Equation (4) reveals a non-closed form. Instead, iterated numerical optimisation algorithms can be used to proxy the roots, most of them considering first or second-order approximations of the function to optimise.

The *Levenberg-Marquardt algorithm* bridges the gap between between the steepest descent method (first-order) and the Newton-Raphson method (second-order) by inflating the diagonal terms of the Hessian matrix. Away from the endpoint, a second-order descent is favoured for its faster convergence pace, while the steepest approach is privileged close to the extremum, as it allows careful refinement of the step size. We use function **marqLevAlg**, since it notably introduces a stringent convergence criteria, the relative distance to the maximum (RDM), which sets apart extrema from spurious saddle points [3].

We provide additional theoretical results, such as analytical formulas for the Gradient and the Hessian in their constrained and unconstrained versions as well as simulation outputs in the vignette of the DeCovarT Github webpage.

2 Simulations

2.1 Simulation of a convolution of multivariate Gaussian mixtures

To assert numerically the relevance of accounting the correlation between expressed transcripts, we designed a simple toy example with two genes and two cell populations. Hence, using the simplex constraint (Eq.(2)), we only have to estimate one free unconstrained parameter, θ_1, and then converts it back to the original ratio space using the mapping function (Eq.(7)).

We simulated "virtual" bulk mixture, $\boldsymbol{y} \in \mathcal{M}_{G \times N}$, for a set of artificial samples $N = 500$, with the following generative model:

- We tested two levels of cellular ratios, one with equi-balanced proportions ($\boldsymbol{p} = (p_1, p_2 = 1 - p_1) = (\frac{1}{2}, \frac{1}{2})$ and one with highly unbalanced cell populations: $\boldsymbol{p} = (0.95, 0.05)$.

[1]The numerical consistency of these derivatives was asserted with the **numDeriv numDeriv** package, using the more stable Richardson's extrapolation

- Then, each purified transcriptomic profile of the two cell populations is drawn from a bivariate Gaussian distribution. We compared two scenarios, playing on the mean distance of centroids, respectively $\mu_{.1} = (20, 22), \mu_{.2} = (22, 20)$ and $\mu_{.2} = (20, 40), \mu_{.2} = (40, 20))$ and building the covariance matrix, $\mathbf{\Sigma} \in \mathcal{M}_{2\times 2}$ by assuming equal individual variances for each gene (the diagonal terms of the covariance matrix, $\text{Diag}(\mathbf{\Sigma_1}) = \text{Diag}(\mathbf{\Sigma_1}) = \boldsymbol{I}_2$) but varying the pairwise correlation between gene 1 and gene 2, $\mathbb{C}\text{ov}\,[x_{1,2}]$, on the following set of values: $\{-0.8, -0.6, \dots, 0.8\}$ for each of the cell population.
- As stated in Eq.(1), we assume that bulk mixture, $\mathbf{y}_{.i}$ could be directly reconstructed by summing up the individual cellular contributions weighted by their abundance, without additional noise.

Precisely, we tested the following general 8 parameters configurations listed in Table below (1) in this bivariate benchmark:

Table 1: The 8 general scenarios tested to compare the performance of DeCovarT vs standard linear deconvolution model

ID	**Entropy**	**OVL**	**Proportions**	**Means**	**Variance**
B1-Ho	1.000	0.1379955	0.5 / 0.5	(20,22);(22,20)	1 / 1
B1-He	1.000	0.1878551	0.5 / 0.5	(20,22);(22,20)	1 / 2
B2-Ho	0.286	0.3193959	0.95 / 0.05	(20,22);(22,20)	1 / 1
B2-He	0.286	0.4310286	0.95 / 0.05	(20,22);(22,20)	1 / 2
B3-Ho	1.000	0.0000000	0.5 / 0.5	(20,40);(40,20)	1 / 1
B3-He	1.000	0.0000000	0.5 / 0.5	(20,40);(40,20)	1 / 2
B4-Ho	0.286	0.0000000	0.95 / 0.05	(20,40);(40,20)	1 / 1
B4-He	0.286	0.0000000	0.95 / 0.05	(20,40);(40,20)	1 / 2

2.2 Results

We compared the performance of DeCovarT algorithm with the outcome of a quadratic algorithm that specifically addresses the unit simplex constraint: the negative least squares algorithm (NNLS, [4]).

Even with a limited toy example including two cell populations characterised only by two genes, we observe that the overlap was a good proxy of the quality of the estimation: the less the two cell distributions overlap, the better the quality of the estimation is (Fig. 2):

The package used to generate the simulations and infer ratios from virtual or real biological mixtures with the DeCovarT algorithm is implemented on personal Github account DeCovarT.

References

[1] T. Erkkilä, S. Lehmusvaara, P. Ruusuvuori, T. Visakorpi, I. Shmulevich, and H. Lähdesmäki, "Probabilistic analysis of gene expression measurements from heterogeneous tissues," *Bioinformatics*, vol. 26, no. 20, pp. 2571–2577, Oct. 2010, doi: 10.1093/bioinformatics/btq406.

[2] R. Mazumder and T. Hastie, "The Graphical Lasso: New Insights and Alternatives," *Electronic Journal of Statistics*, vol. 6, Nov. 2011, doi: 10.1214/12-EJS740.

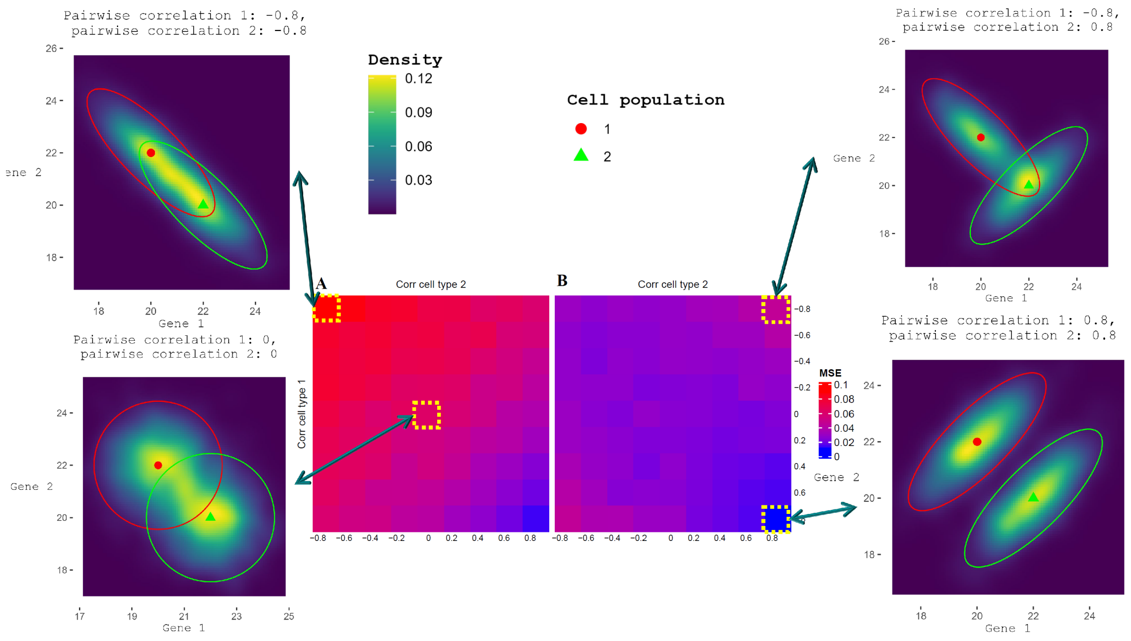


Figure 2: We used the package **ComplexHeatmap** to display the mean square error (MSE) of the estimated cell ratios, comparing the output of the deconRNASEQ algorithm ([**?**]), in Panel **A**, with our newly implemented DeCovarT algorithm, in Panel **B**. The lower the MSE, the least noisy and biased the estimates. n addition, we added two-dimensional density plots of the central scenario, parametrised by a diagonal covariance matrix, and most extreme scenarios, with the highest gene pairwise correlation. The ellipsoids represent for each cell population the 95% confidence region and the red spherical and green triangular shapes represent respectively the centroids of cell population 1 and cell population 2.

[3] M. Prague, D. Commenges, J. Guedj, J. Drylewicz, and R. Thiébaut, “NIMROD: A program for inference via a normal approximation of the posterior in models with random effects based on ordinary differential equations,” *Computer Methods and Programs in Biomedicine*, vol. 111, no. 2, pp. 447–458, Aug. 2013, doi: 10.1016/j.cmpb.2013.04.014.

[4] K. H. Haskell and R. J. Hanson, “An algorithm for linear least squares problems with equality and nonnegativity constraints,” *Mathematical Programming*, vol. 21, no. 1, pp. 98–118, Dec. 1981, doi: 10.1007/BF01584232.

A Appendix A: Theoretical details

First and second-order derivation of the constrained DeCovarT log-likelihood function

To reparametrise the log-likelihood function (Eq.(4)) in order to explicitly handling the unit simplex constraint (Eq.(2)), we consider the following mapping function: $\boldsymbol{\psi} : \boldsymbol{\theta} \to \boldsymbol{p} \mid \quad \boldsymbol{\theta} \in \mathbb{R}^{J-1}$, $\boldsymbol{p} \in]0,1[^J$ (Eq.(7)):

$$\boldsymbol{p} = \boldsymbol{\psi}(\boldsymbol{\theta}) = \begin{cases} p_j = \frac{e^{\theta_j}}{\sum_{k<J} e^{\theta_k} + 1}, \; j < J \\ p_J = \frac{1}{\sum_{k<J} e^{\theta_j} + 1} \end{cases} \qquad \boldsymbol{\theta} = \boldsymbol{\psi}^{-1}(\boldsymbol{p}) = \left(\ln \left(\frac{p_j}{p_J} \right) \right)_{j \in \{1,\dots,J-1\}} \tag{7}$$

that is a C^2-diffeomorphism, since $\boldsymbol{\psi}$ is a bijection between $\boldsymbol{p}$ and $\boldsymbol{\theta}$ twice differentiable.

Its Jacobian, $\mathbf{J}_{\boldsymbol{\psi}} \in \mathcal{M}_{J \times (J-1)}$ is given by Eq.(8):

$$\mathbf{J}_{i,j} = \frac{\partial p_i}{\partial \theta_j} = \begin{cases} \frac{e^{\theta_i} B_i}{A^2}, \quad i = j, \, i < J \\ \frac{-e^{\theta_j} e^{\theta_i}}{A^2}, \quad i \neq j, \, i < J \\ \frac{-e^{\theta_j}}{A^2}, \quad i = J \end{cases} \tag{8}$$

with i indexing vector-valued $\boldsymbol{p}$ and j indexing the first-order order partial derivatives of the mapping function, $A = \sum_{j'<J} e^{\theta_{j'}} + 1$ the sum over exponential (denominator of the mapping function) and $B = A - e^{\theta_i}$ the sum over ratios minus the exponential indexed with the currently considered index i.

The Hessian (which fortunately is symmetric for each component j, as expected according to the Schwarz’s theorem) of the vectorial mapping function $\boldsymbol{\psi}(\boldsymbol{\theta})$ is a third-order tensor of rank $(J-1)(J-1)J$, given by Eq.(9):

$$\frac{\partial^2 p_i}{\partial k \partial j} = \begin{cases} \frac{e^{\theta_i} e^{\theta_l} \left(-B_i + e^{\theta_i}\right)}{A^3}, \; (i < J) \land ((i \neq j) \oplus (i \neq k)) \quad (a) \\ \frac{2 e^{\theta_i} e^{\theta_j} e^{\theta_k}}{A^3}, \; (i < J) \land (i \neq j \neq k) \quad (b) \\ \frac{e^{\theta_i} e^{\theta_j} \left(-A + 2e^{\theta_j}\right)}{A^3}, \; (i < J) \land (j = k \neq i) \quad (c) \\ \frac{B_i e^{\theta_i} \left(B_i - e^{\theta_i}\right)}{A^3}, \; (i < J) \land (j = k = i) \quad (d) \\ \frac{e^{\theta_j} \left(-A + 2e^{\theta_j}\right)}{A^3}, \; (i = J) \land (j = k) \quad (e) \\ \frac{2 e^{\theta_j} e^{\theta_k}}{A^3}, \; (i = J) \land (j \neq k) \quad (f) \end{cases} \tag{9}$$

with i indexing $\boldsymbol{p}$, j and k respectively indexing the first-order and second-order partial derivatives of the mapping function with respect to $\boldsymbol{\theta}$. In line (a), $\oplus$ refers to the Boolean XOR operator, $\land$ to the AND operator and $l = \{j,k\} \setminus i$.

To derive the log-likelihood function in Eq.(5), we reparametrise $\boldsymbol{p}$ to $\boldsymbol{\theta}$, using a standard *chain rule formula*). Considering the original log-likelihood function, Eq.(4), and the mapping function, Eq.(7), the differential at the first order and at the second order is given by Eq.(10) and Eq.(11), respectively defined in $\mathbb{R}^{J-1}$ and $\mathcal{M}_{(J-1)\times(J-1)}$:

$$\left[\frac{\partial \ell_{\boldsymbol{y}|\boldsymbol{\zeta}}}{\partial \theta_j}\right]_{j<J} = \sum_{i=1}^{J} \frac{\partial \ell_{\boldsymbol{y}|\boldsymbol{\zeta}}}{\partial p_i} \frac{\partial p_i}{\partial \theta_j} \tag{10}$$

$$\left[\frac{\partial \ell^2_{\boldsymbol{y}|\boldsymbol{\zeta}}}{\partial \theta_k \theta_j}\right]_{j<J,\,k<J} = \sum_{i=1}^{J} \sum_{l=1}^{J} \left(\frac{\partial p_i}{\partial \theta_j} \frac{\partial^2 \ell_{\boldsymbol{y}|\boldsymbol{\zeta}}}{\partial p_i \partial p_l} \frac{\partial p_l}{\partial \theta_k} \right) + \sum_{i=1}^{J} \left(\frac{\partial \ell_{\boldsymbol{y}|\boldsymbol{\zeta}}}{\partial p_i} \frac{\partial^2 p_i}{\partial \theta_k \theta_j} \right) \quad (d) \tag{11}$$