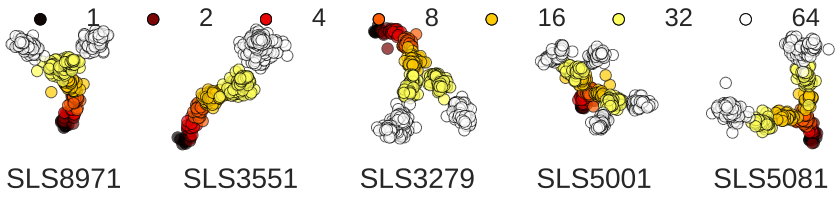

Figure S1: **Differentiation Process Simulations** Simulated differentiation processes along cell stages. The cell stages are coloured from 1 to 64 cell stage and each simulation has its associated unique seed printed underneath. The selection of differentiation processes was done by visual inspection, sifting for variety and non overlapping profiles, so that a 2 dimensional landscape was possible.

S1 Simulation Study

The code is available via github at the following address. `https://github.com/mzwiessele/topslam`. The simulation study can also be found there, as well as plotting code for plotting Waddington's landscape using python and matplotlib. All code is based upon GPy [1], the Gaussian Process toolbox for python.

S1.1 Differentiation Profile Simulations

We simulated several differentiation profiles to test all methods against each other.

Simulation was done by simulating a branching process in two dimensions with increasing probability of splitting in a cell stage along the time line. This ensures a random pattern of splitting. The random nature of the splitting process made it necessary to select simulated landscapes by hand (repeatedly simulating and selecting by visual inspection). Now we can simulate gene expression profiles by sampling from independent Gaussian processes with differing priors for 48 different genes. The priors for the mapping include a linear mapping and both long-term and short-term non-linear trends to simulate the complexity of biological processes. This makes the simulations more complex than any of the models we applied to recover the data. The inputs to these gene expression profiles are composed of the simulated differentiation processes and the simulated pseudo time driving the process.

To ensure a similar distance structure in the simulations to the real data, we compare the distribution over squared distances. We want the underlying structure to be similar to the one seen in the real world. In this we found RNA-seq experimental data to be very different from the data received by qPCR. See the squared distance distributions plotted in Figure S2. As the RNASeq experiments only have one split in cell types, the Waddington landscape can be explained by a linear embedding in both cases and were excluded from the main paper.

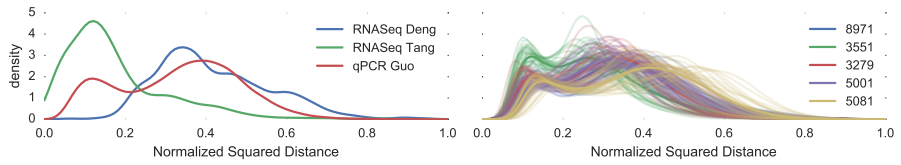

Figure S2: Distribution of squared distance in the three different data sets (left) and the simulated gene expression profiles (right). The left plot shows distributions of distances for one qPCR experiment of [2] and two RNASeq experiments [3, 4]. In the right plot we see all traces of the 10 repetitions and the mean estimate as thick line. Note the difference in distances between RNA-seq and qPCR experiments. Our simulation simulates the distances of the qPCR experiments. Shown are the kernel density estimates for the upper triangular of the squared distances, normalised in the range $[0, 1]$, for comparability.

The code for the simulation can be found in the python package provided (above).

S1.2 Extraction and Comparison of Landscapes

We applied different dimensionality reduction techniques to these simulated latent spaces and calculated the Pearson correlation coefficients ρ between simulated time and extracted pseudo time for 10 independent high dimensional data sets per simulated latent space. We plotted the extracted landscape by topslam and linear regression between simulated and extracted times in Figures S3 and S4.

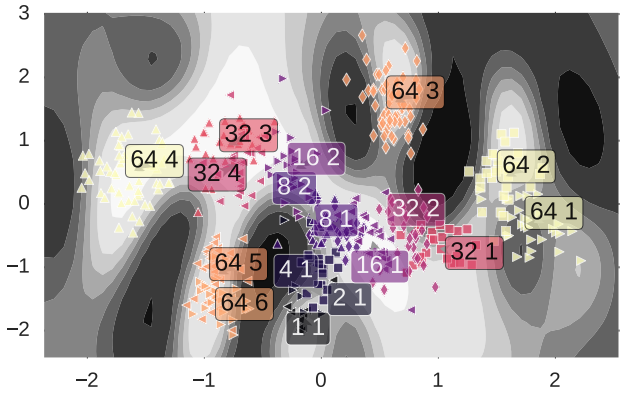

Figure S3: **Pseudo Time Extraction Tree** Example data with the tree along which we can see the extracted times depicted as the colour of the edges in arbitrary pseudo times from dark blue to light blue. The shading depicts Waddington's landscape from favoured areas (light) to un-favoured areas (dark).

First, we extract the ordering information using the MST approach, building a MST and follow its shortest paths to order the cells seen along it. Results for

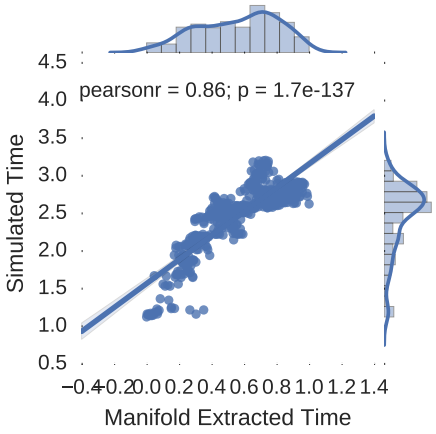

Figure S4: **Pseudo Time Comparison** Regression (and regression coefficient) of extracted pseudo time versus the simulated time in the simulation.

all dimensionality reduction techniques are shown in Figure S5 and Table S1.

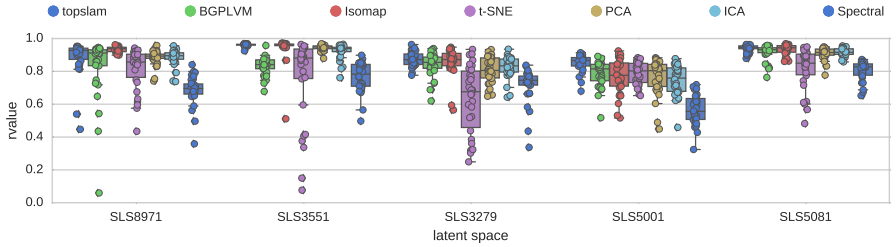

Figure S5: Simulation study results for gene expression matrices generated from simulated Waddington landscapes along a time line. Shown are linear regression Pearson correlation coefficients ρ between extracted and simulated time lines. Extraction was done by creating a 3-nearest-neighbour graph, while embedding the MST for a fully connected graph. Data sets were simulated from different differentiation profiles as described in Section S1.

S1.3 Waddington's Landscape for other Dimensionality Reduction Techniques

Our method is not restricted to learn the landscape fully by itself. It can use other dimensionality reduction techniques as bases and overlay a probabilistic landscape over it. We first show that overlaying probabilistic Waddington's landscapes over existing methods does not decrease performance. Second, we will show how to improve existing methods by learning probabilistic Waddington's landscapes from existing methods.

	SLS8971	SLS3551	SLS3279	SLS5001	SLS5081
topslam	0.88±0.11	**0.96**±0.01	**0.87**±0.04	**0.85**±0.05	**0.94**±0.02
BGPLVM	0.83±0.19	0.83±0.06	0.84±0.07	0.79±0.08	0.92±0.05
Isomap	**0.93**±0.02	0.94±0.08	0.86±0.09	0.77±0.11	0.93±0.03
t-SNE	0.82±0.13	0.76±0.26	0.63±0.21	0.79±0.07	0.82±0.13
PCA	0.88±0.05	0.94±0.02	0.81±0.08	0.76±0.11	0.91±0.04
ICA	0.88±0.05	0.92±0.05	0.82±0.07	0.75±0.09	0.92±0.02
Spectral	0.68±0.09	0.77±0.09	0.71±0.11	0.57±0.09	0.81±0.06

Table S1: Table for the tree extracted pseudo time simulations from Figure S5. Results are shown as mean and standard deviations for Pearson correlation coefficients ρ between simulated and extracted pseudo times.

S1.3.1 Overlay Landscape

We can retrospectively apply our landscape reconstruction on top of other dimensionality reduction methods. By taking the inferred latent positions of the cells to be the correct locations for the Bayesian GPLVM S3 model we can overlay our method to extract the relevant metric information from the resulting probabilistic Waddington's landscape. In technical terms, we fix the latent space of topslamto be the latent space learnt by the other techniques and overlay the probabilistic Waddington's landscape on top of it.

We compare all the pseudotimes extracted from the landscapes produced by other methods to the probabilistic Waddington's landscapes extracted by combining our method with the landscape of the other methods. We note that by doing so we only increase the correlation between the estimates and the simulated ground truth, performance is never decreased (Figure S6).

S1.3.2 Learn Landscape

Just overlaying a landscape over existing methods does not decrease performance in recovering the intrinsic signals of the cells. We can improve upon this, by allowing our method to fully learn a probabilistic Waddington's landscape from other methods as initial estimates. We initialise our method with an existing technique and let it learn the landscape from there. In other words, we do not fix the latent spaces of the other techniques, but rather we learn a new latent space using the original points as an initialisation.

This approach results in significant improvement of both correlation to simulated times, as well as robustness to data generation. We have summarised all results of the landscape correction using topslamin Figure S7.

Using our method significantly improves the performance for recovery of simulated pseudotimes over existing techniques. This comes from the probabilistic modelling of the landscape and correction for topographical distortions.

We could show that taking the landscapes topography into account significantly improves recovery of intrinsic signals in the measurements.

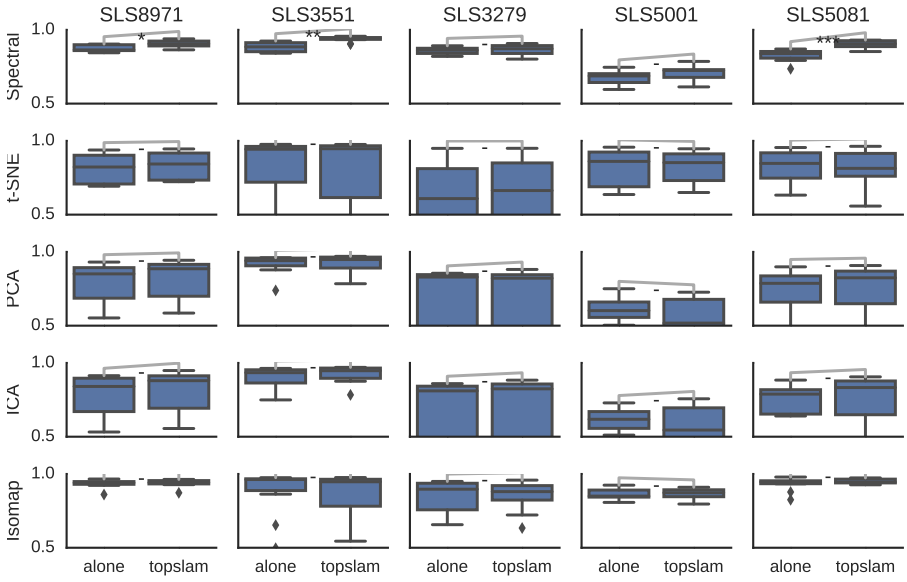

Figure S6: Comparison of correlation between pseudotimes extracted from other dimensionality reduction techniques landscapes and a correction of those techniques with our metric derived from the probabilistic Waddington's landscape. To the left of each subplot you see the pure landscape (*alone*) and to the right is our corrected landscape.

S2 High-Throughput Single-Cell Gene Expression Measurement

S2.1 Targeted

Quantitative Polymerase Chain Reaction (qPCR) is a directed way of measuring the expression of previously known genes in down to singular cells. The transcribed mRNA of a cell to measure is usually not enough for detection, and thus, has to be amplified. The measurement of expression is done in a fixed amount of reverse detection steps (usually 40, of which the ones with a detection > 28 get set to 28; this ensures a $> 99\%$ chance of detecting the expression of genes which are expressed < 28). As soon as we can detect a hybridisation of the mRNA molecules of question, we can assign the number of times the amplification took place as the expression level, usually referred to as a value in Ct.

For this technique the genes of interest have to be known in advance and we need two sided primers for the PCR amplification. This limits the number of measured genes immensely, but it makes dropout events less likely. As the primer mixes are highly specific, dropout events are not much of an issue, except for very low numbers of DNA molecules ($< 5 - 10$). The noise introduced by this method is due to these saturation effects which can be modelled by a probit likelihood, as done by [5], which treats expression values higher then 28 Ct as outliers.

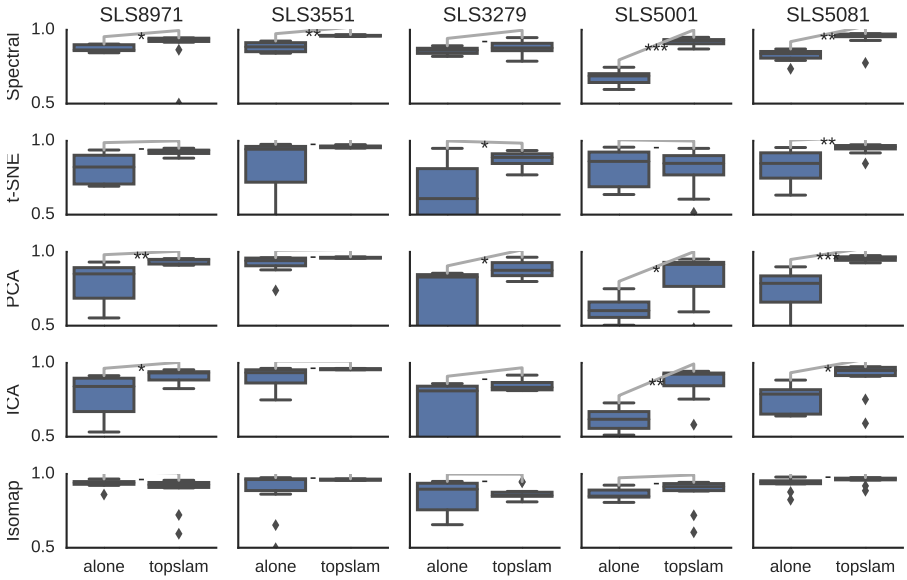

Figure S7: Compare correlation between pseudotimes extracted from other dimensionality reduction techniques landscapes and learning of probabilistic Waddington's landscape from those landscapes. This corrects the landscapes of the other techniques and uses them as a starting point (initialisation).

S2.2 Genome Wide

To measure the genome wide gene expression of genes of single cells we need to make use of high-throughput RNA counting techniques, such as digital PCR, as described in [6]. The translated RNA gets back-translated to DNA and amplified by limiting dilution. The DNA molecules get partitioned into cells, such that each cell is expected to have one or less molecule inside. The cells then get amplified using one PCR per cell and the gene expression is the count of detected DNA molecules, which can be sequenced and mapped back to the respective genes. This technique is called Single-cell tagged reverse transcription (RT-PCR, [6]).

As the genome wide gene expression measurement of single cells is more prone to dropout events usually more strict filtering techniques are required to minimise the intrinsic noise of biological processes and technical variation in the experiment. This usually means to filter genes with low variance or low mean expression levels.

In short: The targeted nature of qPCR makes this method less prone to dropout events in low expressed genes, but there is a expression detection cap at the high end, making it difficult to differentiate between highly expressed genes. RT-PCR allows for genome wide gene expression measurements, leading to low to mid expressed genes having a 50% chance of a dropout event.

S3 Bayesian Gaussian Process Latent Variable Model

In this section we will describe the probabilistic method for manifold learning and correction. We used a variational extension to Gaussian Processes ($\mathcal{GP}$s), in which we variationally integrate out the latent space (i.e. the latent landscape) to learn the most likely inputs for the given observed variables (such as gene expression).

S3.1 Notation

Let our data be composed of N input, output pairs $(\mathbf{x}, \mathbf{y})_{i=0}^{N}$, such that we can pull together the inputs $\mathbf{X} \in \mathbb{R}^{N \times Q}$ and outputs $\mathbf{Y} \in \mathbb{R}^{N \times D}$ into matrices. $\mathcal{GP}$s rely on the pairwise covariance matrix $\mathbf{K}_{ij}$ between all pairs of inputs defined as

$$\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j),$$

where k is a semi-positive definite[7] function, creating a semi-positive definite kernel matrix $\mathbf{K}$. A $\mathcal{GP}$ uses the covariance structure between $\mathbf{X}$ pairs $\mathbf{K}_{\mathbf{FF}}$ in order to learn a latent function $\mathbf{F}$, with prior $\mathbf{F} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{FF}})$ which is integrated with the likelihood for the data $\mathbf{Y} \sim \mathcal{N}(\mathbf{F}, \sigma^2 \mathbf{I})$ as

$$p(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{F}) p(\mathbf{F}|\mathbf{X}) \mathbf{F} \quad .$$

In the following we define several kernel matrices between different inputs by sub indexing with the corresponding latent functions they comprise, e.g. $\mathbf{K}_{\mathbf{FF}}$ is the covariance matrix between inputs $\mathbf{X}$, which describe the latent function $\mathbf{F}$; $\mathbf{K}_{\mathbf{FU}}$ is the covariance matrix describing the covariance between $\mathbf{F}$ and $\mathbf{U}$, where $\mathbf{U}$ is another latent function, see next section.

S3.2 Sparse $\mathcal{GP}$s

Sparse $\mathcal{GP}$s to speed up computation from $\mathcal{O}(N^3)$ to $\mathcal{O}(NM^2)$. Introduce inducing inputs (which act as approximated input points for the covariance of the whole $\mathcal{GP}$) $\mathbf{Z} \in \mathbb{R}^{M \times Q}$, where $M << N$, with corresponding latent function values $\mathbf{U} \in \mathbb{R}^{M \times D}$, defined by the $\mathcal{GP}$ covariance function. The distribution of those is easily defined through the following expressions:

$$\begin{aligned} p(\mathbf{F}|\mathbf{U}) &= \mathcal{N}\left(\mathbf{F}|\mathbf{K}_{\mathbf{FU}}\mathbf{K}_{\mathbf{UU}}^{-1}\mathbf{U}, \mathbf{\Lambda}\right) \\ \mathbf{\Lambda} &= \mathbf{K}_{\mathbf{FF}} - \mathbf{K}_{\mathbf{FU}}\mathbf{K}_{\mathbf{UU}}^{-1}\mathbf{K}_{\mathbf{UF}} \\ p(\mathbf{Y}|\mathbf{F}) &= \mathcal{N}\left(\mathbf{Y}|\mathbf{F}, \sigma^2\mathbf{I}\right) \quad . \end{aligned}$$

Let $\beta = \sigma^{-2}$ in the following to remove some clutter. Then we can integrate out the latent function values of the $\mathcal{GP}$ prior $\mathbf{f}$

$$\begin{aligned} \log p(\mathbf{Y}|\mathbf{U}) &= \log \langle p(\mathbf{Y}|\mathbf{F}) \rangle_{p(\mathbf{F}|\mathbf{U})} \\ &\geq \langle \log p(\mathbf{Y}|\mathbf{U}, \mathbf{F}) \rangle_{p(\mathbf{F}|\mathbf{U})} \quad \text{Jensen's bound} \\ &= \log \mathcal{N}\left(\mathbf{Y}|\mathbf{M}_{\mathcal{L}_1}, \beta^{-1}\mathbf{I}\right) - \tfrac{1}{2}\beta \operatorname{tr} \mathbf{\Lambda} =: \mathcal{L}_1 \quad , \\ \mathbf{M}_{\mathcal{L}_1} &= \mathbf{K}_{\mathbf{FU}}\mathbf{K}_{\mathbf{UU}}^{-1}\mathbf{U} \quad . \end{aligned}$$

We use the $\mathcal{L}_1$-bound to approximate the true $\mathcal{GP}$ posterior through the latent $\mathcal{GP}$ function values $\mathbf{U}$

$$\begin{aligned}\log p(\mathbf{Y}|\mathbf{X},\mathbf{Z}) &= \log \langle p(\mathbf{Y}|\mathbf{U})\rangle_{p(\mathbf{U})} \\ &\geq \log \langle \exp \mathcal{L}_1 \rangle_{p(\mathbf{U})} \\ &= \log \mathcal{N}\left(\mathbf{Y}|\mathbf{0}, \mathbf{K}_{\mathbf{FU}}\mathbf{K}_{\mathbf{UU}}^{-1}\mathbf{K}_{\mathbf{UF}} + \beta^{-1}\mathbf{I}\right) \\ &\quad - \tfrac{1}{2}\beta \operatorname{tr}\mathbf{\Lambda} =: \mathcal{L}_2 \ .\end{aligned}$$

Thus, we can optimise the sparse $\mathcal{GP}$ bound $\mathcal{L}_2$ with respect to the positions of the inducing inputs $\mathbf{Z} \in \mathbb{R}^{M\times Q}$ in $\mathcal{O}(NM^2)$. This computation is dominated by the computation of $\mathbf{K}_{\mathbf{FU}}\mathbf{K}_{\mathbf{UU}}^{-1}\mathbf{K}_{\mathbf{UF}}$, where the inversion of the inducing inputs covariance matrix $\mathbf{K}_{\mathbf{UU}}^{-1}$ can be computed in $\mathcal{O}(M^3)$, which is dominated by the product between $\mathbf{K}_{\mathbf{FU}}\mathbf{K}_{\mathbf{UU}}^{-1}$, thus the complexity of the algorithm is $\mathcal{O}(NM^2)$.

S3.3 Variational Bound for Latent Space Integration

In the next step we want to integrate out the inputs $\mathbf{X}$, to provide a latent variable model, where the latent space $\mathbf{X}$ is learnt as the solution to the mapping $f(\mathbf{X}) = \mathbf{Y}$. $f(\mathbf{X})$ has a GP prior and $p(\mathbf{X})$ gets variationally estimated using a unit variance Gaussian prior $q(\mathbf{X}) = \mathcal{N}(\mathbf{M}, \mathbf{S})$, where $\mathbf{S}$ contains the diagonals for each dimension as columns, such that the corresponding variance for $\mathbf{M}_{\cdot 0}$ is $\operatorname{diag}\mathbf{S}_{\cdot 0}$. In the following we omit the inducing inputs $\mathbf{Z}$ from the conditioning to avoid cluttering the expressions.

$$\begin{aligned}\log p(\mathbf{Y}) &= \log \int p(\mathbf{Y}|\mathbf{X})p(\mathbf{X}) \ \mathrm{d}\mathbf{X} \\ &= \log \int p(\mathbf{Y}|\mathbf{X})p(\mathbf{X})\frac{q(\mathbf{X})}{q(\mathbf{X})} \ \mathrm{d}\mathbf{X} \\ &= \log \left\langle \frac{p(\mathbf{Y}|\mathbf{X})p(\mathbf{X})}{q(\mathbf{X})} \right\rangle_{q(\mathbf{X})} \\ &\geq \left\langle \log p(\mathbf{Y}|\mathbf{X}) - \log \frac{q(\mathbf{X})}{p(\mathbf{X})} \right\rangle_{q(\mathbf{X})} \\ &\geq \langle \mathcal{L}_2 \rangle_{q(\mathbf{X})} - \mathrm{KL}\left(q(\mathbf{X}) \| p(\mathbf{X})\right) \\ &=: \mathcal{L}_3 \ .\end{aligned}$$

Thus, when maximising the above bound $\mathcal{L}_3$ we will minimise the distance between p and q, while maximising the sparse GP bound, which will try to find a suitable fit for the latent function $\mathbf{F}$ to fit the seen data $\mathbf{Y}$, at the same time as finding the right input locations $\mathbf{X}$ to provide the most likely latent space under the current function $\mathbf{F}$. This can be confusing in the beginning, we will see the different contributions for the latent space later on.

Let us focus on the left side of the expression. We will use $\mathcal{L}_2$ to extract the variational bound for Bayesian treatment of the latent space $\mathbf{X}$. With $\langle\cdot\rangle = \langle\cdot\rangle_{q(\mathbf{X})}$

$$\begin{aligned}\langle \mathcal{L}_2 \rangle = &- \tfrac{ND}{2}\log 2\beta\pi + \tfrac{D}{2}\log|\mathbf{K}_{\mathbf{UU}}| + \tfrac{1}{2}\log|\mathbf{K}_{\mathbf{UU}} + \beta\mathbf{\Psi}_2| \\ &- \tfrac{1}{2}\operatorname{tr}\mathbf{Y}^\top(\beta\mathbf{I} - \beta^2\mathbf{\Psi}_1(\mathbf{K}_{\mathbf{UU}} + \beta\mathbf{\Psi}_2)^{-1}\mathbf{\Psi}_1^\top)\mathbf{Y} \\ &- \tfrac{\beta}{2}\Psi_0 + \tfrac{\beta}{2}\operatorname{tr}(\mathbf{K}_{\mathbf{UU}}^{-1}\mathbf{\Psi}_2) \ ,\end{aligned}$$

where

$$\Psi_0 = \langle \operatorname{tr} \mathbf{K}_{\mathbf{FF}} \rangle \qquad \mathbf{\Psi}_1 = \langle \mathbf{K}_{\mathbf{FU}} \rangle \qquad \mathbf{\Psi}_2 = \sum_{n=1}^{N} \langle k(\mathbf{x}_n, \mathbf{z}) k(\mathbf{z}', \mathbf{x}_n) \rangle$$

S4 Pseudotime Extraction

Extracting pseudotime information from the latent space embedding provided by Bayesian GPLVM is done by constructing a distance matrix between all pairs of single cell embeddings $\mathbf{X}$.

$$D_{ij}^{\text{latent}} = \sqrt{(\mathbf{X}_i - \mathbf{X}_j)^\top (\mathbf{X}_i - \mathbf{X}_j)} \tag{S1}$$

These distances are corrected using the Wishart embedding metric tensor as described by [8]. By extracting a Wishart distribution $\mathcal{W}(p, \mathbf{\Sigma}, \boldsymbol{\mu}^\top \boldsymbol{\mu})$ defined on the Jacobian $\frac{\partial p(\mathbf{Y})}{\partial \mathbf{X}} = \mathbf{J}$ of the latent embedding $p(\mathbf{Y}|\mathbf{X})$. $\boldsymbol{\mu}$ is the mean of the gradient of the posterior distribution of the GP, and $\mathbf{\Sigma}$ is its covariance. We compute the expected manifold metric tensor,

$$\mathbf{M} \coloneqq \left\langle \mathbf{J}^\top \mathbf{J} \right\rangle = \langle \mathbf{J} \rangle^\top \langle \mathbf{J} \rangle + D \operatorname{cov}(\mathbf{J}, \mathbf{J}) \tag{S2}$$

$$= \boldsymbol{\mu}^\top \boldsymbol{\mu} + D\mathbf{\Sigma}. \tag{S3}$$

This metric tensor describes the distortions of the space. Think of a sphere, where we can describe every point on its surface with two coordinates, but itself lives in a three dimensional space. In order to correct for latent space embeddings, we would need to follow the geodesics of the latent space embedding (lines with shortest distance between points along the manifold/sphere), which is expensive to compute in the Bayesian GPLVM embedding. Therefore, we use a heuristic approach to correct for the embedding distortions. We correct locally for distortions of pairs of latent embeddings of cells, such that local distances are corrected for

$$D_{ij}^{\text{corrected}} = \sqrt{(\mathbf{X}_i - \mathbf{X}_j)^\top \frac{\mathbf{M}_i + \mathbf{M}_j}{2} (\mathbf{X}_i - \mathbf{X}_j)} \ .$$

We now use these corrected distances to construct a minimal spanning tree and compute distances along this tree (Figure S3), using for example Dijkstras algorithm for shortest paths along a graph. With that, the distances computed follow approximately the geodesics and can be used to extract pseudo time orderings for the cells (Figure S4). We show the process in a simulation application to a manifold embedding in Section S1.

S4.1 Waddington's Landscape

In order to show a representation of Waddingon's landscape, we use the so called magnification factor of the expected manifold metric tensor

$$\mathbf{W}_i \coloneqq \sqrt{\det \mathbf{M}_i} \ \ \forall 1 \leq i \leq N \ . \tag{S4}$$

To plot the representation of Waddington's landscape in three dimensions, we zero mean and normalise the expected manifold metric tensor variance to 2 and

push it through a logit function to better show hills and valleys of the surface. See `https://github.com/mzwiessele/topslam` for the implementation of these features (and an example Jupyter notebook to show application).

S4.2 Caveats

The method presented relies on the correct fitting of the Bayesian GPLVM approach. Bayesian GPLVM is a variational extension of the standard GPLVM [9]. It treats the latent space embedding $\mathbf{X}$ with a probabilistic prior $p(\mathbf{X}) = \mathcal{N}(\mathbf{X}|\mathbf{mu}_\mathbf{X}, \mathbf{\Sigma}_\mathbf{X})$. This provides an extremely general model and, thus it can lead to an optimisation surface with local optima. Thus, it might present difficulties to optimise a Bayesian GPLVM. We provide in the package some techniques for creating and optimising a Bayesian GPLVM.

In the supplied package we provide functions for filtering data optimally for usage in model learning and optimisation routines, which usually help to improve the learnt landscapes. Additionally we supply all plotting code to be able to reproduce landscapes as shown here.

S4.3 Supplementary Figures

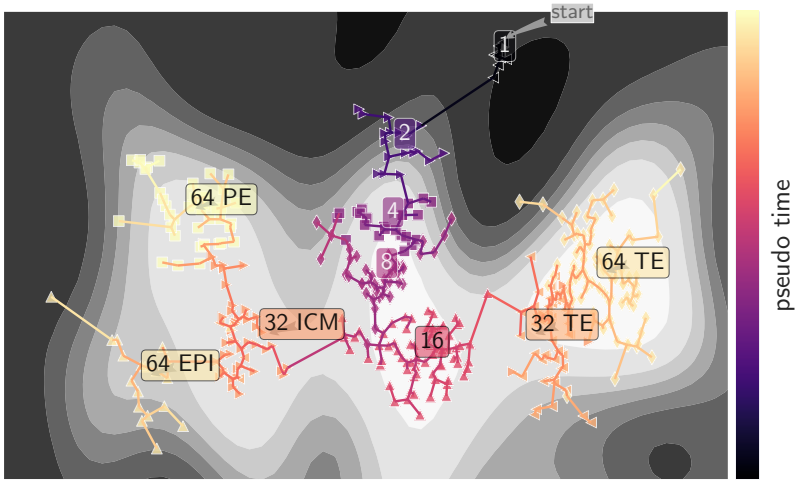

Figure S8: The pseudo time extracted by the probabilistic handling of Waddington's landscape from a qPCR single cell experiment over mouse embryonic stem cells. You can see the time progression along the cells depicted by the colouring of the branches of the spanning tree. The labels for the cells are extracted using a nested PCA approach, filtering at each step further (See [2] for details). The shading shows Waddington's landscape as grey scales, from dark (un-favoured) to light (favoured).

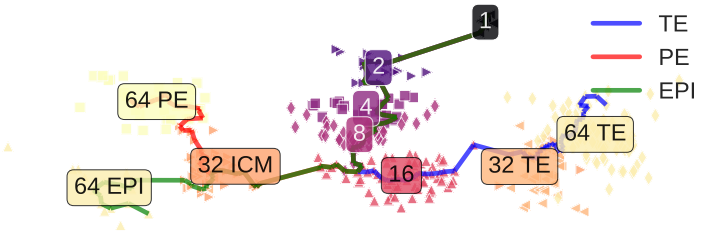

Figure S9: Differentiation process along the time graph (endpoints randomly chosen within respective cell type). These differentiation paths are used for differential gene expression for marker gene detection.

References

[1] GPy. GPy: A Gaussian process framework in python. `http://github.com/SheffieldML/GPy` (since 2012).

[2] Guo, G. *et al.* Resolution of cell fate decisions revealed by single-cell gene expression analysis from zygote to blastocyst. *Developmental cell* **18**, 675–685 (2010).

[3] Tang, F. *et al.* mRNA-Seq whole-transcriptome analysis of a single cell. *Nature methods* **6**, 377–382 (2009).

[4] Deng, Q., Ramsköld, D., Reinius, B. & Sandberg, R. Single-Cell RNA-Seq Reveals Dynamic, Random Monoallelic Gene Expression in Mammalian Cells. *Science* **343**, 193–196 (2014). URL `http://www.sciencemag.org/content/343/6167/193.abstract`. `http://www.sciencemag.org/content/343/6167/193.full.pdf`.

[5] Buettner, F. & Theis, F. J. A novel approach for resolving differences in single-cell gene expression patterns from zygote to blastocyst. *Bioinformatics* **28**, i626–i632 (2012).

[6] Kalisky, T. & Quake, S. R. Single-cell genomics. *Nat Meth* **8**, 311–314 (2011). URL `http://dx.doi.org/10.1038/nmeth0411-311`.

[7] Williams, C. K. & Rasmussen, C. E. *Gaussian processes for machine learning*, vol. 2 (2006).

[8] Tosi, A., Hauberg, S., Vellido, A. & Lawrence, N. D. Metrics for probabilistic geometries. In *Proceedings of 30th Conference on Uncertainty in Artificial Intelligence (uai 2014)* (AUAI Press Corvallis, 2014).

[9] Lawrence, N. Probabilistic Non-linear Principal Component Analysis with Gaussian Process Latent Variable Models. *Journal of Machine Learning Research* (2005).