Bioinformatics, 37(19), 2021, 3252–3262
doi: 10.1093/bioinformatics/btab260
Advance Access Publication Date: 11 May 2021
Original Paper

Gene expression

RVAgene: generative modeling of gene expression time series data

Raktim Mitra and Adam L. MacLean *

University of Southern California, Los Angeles, CA 90007, USA

*To whom correspondence should be addressed.

Associate Editor: Anthony Mathelier

Received on November 17, 2020; revised on April 19, 2021; editorial decision on April 20, 2021; accepted on April 22, 2021

Abstract

Motivation: Methods to model dynamic changes in gene expression at a genome-wide level are not currently sufficient for large (temporally rich or single-cell) datasets. Variational autoencoders offer means to characterize large datasets and have been used effectively to characterize features of single-cell datasets. Here, we extend these methods for use with gene expression time series data.

Results: We present RVAgene: a recurrent variational autoencoder to model gene expression dynamics. RVAgene learns to accurately and efficiently reconstruct temporal gene profiles. It also learns a low dimensional representation of the data via a recurrent encoder network that can be used for biological feature discovery, and from which we can generate new gene expression data by sampling the latent space. We test RVAgene on simulated and real biological datasets, including embryonic stem cell differentiation and kidney injury response dynamics. In all cases, RVAgene accurately reconstructed complex gene expression temporal profiles. Via cross validation, we show that a low-error latent space representation can be learnt using only a fraction of the data. Through clustering and gene ontology term enrichment analysis on the latent space, we demonstrate the potential of RVAgene for unsupervised discovery. In particular, RVAgene identifies new programs of shared gene regulation of *Lox* family genes in response to kidney injury.

Availability and implementation: All datasets analyzed in this manuscript are publicly available and have been published previously. RVAgene is available in Python, at GitHub: https://github.com/maclean-lab/RVAgene; Zenodo archive: http://doi.org/10.5281/zenodo.4271097.

Contact: macleana@usc.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Dynamic changes in gene expression control the transcriptional state of a cell, and are responsible for modulating cellular states and fates. Gene expression dynamics are in turn controlled by cell-internal and external signaling networks. Despite the noisiness of gene expression in single cells (Raj and Van Oudenaarden, 2008), over time or over populations of cells, predictable patterns emerge. Here, we address the challenge of classifying and predicting gene expression dynamics across large groups of genes.

Machine learning (and deep learning in particular) has led to recent advances in our ability to explain or predict biological phenomena (Ching *et al.*, 2018). Deep learning modeling via autoencoders (Hinton and Salakhutdinov, 2006) and variational autoencoders (Kingma and Welling, 2014) has been central to progress in the field. Autoencoders learn two functions: one to encode each input data point to a low dimensional point, and another (the decoder) to reconstruct the original data point from the low dimensional representation. Variational autoencoders (VAEs) build on this architecture and instead encode input data points as distributions; VAEs are less prone to overfitting and can offer meaningful representations of biological features in the latent space (Way and Greene, 2017).

Single-cell mRNA sequencing (scRNA-seq) data present appealing sources of data for deep learning models, given their size and complexity (Svensson *et al.*, 2018). Deep learning models have been used to analyze scRNA-seq data and address a variety of challenges. Autoencoders have been developed to perform noise removal/batch correction (Deng *et al.*, 2019; Eraslan *et al.*, 2019; Wang *et al.*, 2019), imputation (Talwar *et al.*, 2018) and visualization and clustering (Lin *et al.*, 2017). VAEs have been developed for the visualization and clustering of scRNA-seq data (Ding *et al.*, 2018; Wang and Gu, 2018), and can provide a broad framework for generative modeling of scRNA-seq data (Lopez *et al.*, 2018): scVI can be used for batch correction, clustering, visualization and differential expression testing.

The methods described above for single-cell data analysis by deep learning focus primarily on cell-centric tasks; here, we are interested in gene-centric inference. Particularly, we are interested in characterizing dynamic changes in gene expression. These can be either changes with respect to real time or 'pseudotime,' the latter referring to the ordering of single cells along an axis describing a dynamic cell process such as development or stem cell differentiation (see methods overview in (Saelens *et al.*, 2019)). We can interpret any scRNA-seq data as gene expression time series data, given an appropriate underlying temporal process, either in terms of real (experimental) time (low resolution: around 2–20 data points) or pseudotime (high resolution: $10^3 - 10^6$ data points). McDowell *et al.* (2018) introduced a non-parametric hierarchical Bayesian method (DPGP) to model such data. Using a Gaussian process to cluster temporal gene profiles and a Dirichlet process to generate the Gaussian processes, DPGP offers powerful and intuitive means with which to cluster gene expression time series data. However, since learning Gaussian processes is equivalent to a fully agnostic search in function space, training DPGP is computationally intensive and difficult to parallelize.

Clustering relies on strong assumptions about the underlying structure of the data. Even for methods that move away from hard clustering toward probabilistic methods for cell type assignment (Jetka *et al.*, 2018; Zhu *et al.*, 2019), assumptions remain and under certain conditions a continuous representation of the data may be better. Here, we take such an approach, and seek to find a low dimensional representation of the data, on which further analyses (including but not limited to clustering) can be performed. VAEs are an obvious choice, given their success on other scRNA-seq analysis tasks, but modeling temporal changes with a feed-forward VAE would be equivalent to a fully agnostic search, similar to learning a Gaussian process. Recurrent networks offer well-established architectures for learning sequential and temporal data, and have been successfully combined with VAEs (Fabius and van Amersfoort, 2014). We use a recurrent network architecture to take advantage of the structure in the data.

We introduce a recurrent variational autoencoder for modeling gene dynamics from scRNA-seq data (RVAgene). RVAgene learns two functions during training, parameterized by encoder and decoder networks. The encoder network projects the training data into latent space (we use a 2 or 3 dimensions in order to visualize, though there are no inherent limits). The decoder network learns a reconstruction of training genes from their latent representation. RVAgene facilitates clustering or other characterization of gene profiles in the latent space. By sampling points from the latent space and decoding them, RVAgene provides means to generate new gene expression time series data, drawn from the biological process that was encoded. Overall, RVAgene serves as a multipurpose generative model for exploring gene expression time-series data.

The remainder of the article is structured as follows: we next present methodological details and development of RVAgene. We produce a synthetic gene expression time-series dataset with innate cluster structure, and demonstrate the accuracy of RVAgene on these data. We then explore two biological datasets with RVAgene: a scRNA-seq dataset on stem cell differentiation over pseudotime, on which we demonstrate the advantages of RVAgene over alternative approaches; and a bulk RNA-seq dataset describing dynamic responses to kidney injury, on which we demonstrate the potential for biological discovery. We also present evidence for the efficiency and scalability of RVAgene, and we conclude by discussing its key features and limitations, in light of recent advances in machine learning that will pave the way for future work in these directions.

2 Materials and methods

We develop a recurrent variational autoencoder to model gene expression dynamics (RVAgene). Here, we briefly describe the methods underpinning variational autoencoders, and present the implementation of RVAgene.

2.1 Variational inference and variational autoencoders

In the most general setting of a Bayesian model, we seek to learn the latent variables $\mathbf{z}$ that best characterize some data $\mathbf{x}$. Given a generative process that draws latent variables from a prior distribution, $p(\mathbf{z})$, and a likelihood of the data observed that is given by $p(\mathbf{x}|\mathbf{z})$, then the posterior probability is given by Bayes rule:

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{\int_z p(\mathbf{x}|\mathbf{z})p(\mathbf{z})dz}. \tag{1}$$

The denominator is often intractable, making it difficult to estimate $p(\mathbf{z}|\mathbf{x})$. Markov Chain Monte Carlo methods provide means to estimate posterior probability distributions. An alternative method to estimate hard-to-compute probability distributions is Variational Inference (VI) (Hoffman *et al.*, 2013), which starts from the assumption that the posterior can be approximated by a distribution $q(\mathbf{z})$ from the family $\mathcal{Q}$. VI then amounts to an optimization problem to find the q^* that minimizes the Kullback–Leibler (KL) divergence between the approximation and the true posterior:

$$q^*(\mathbf{z}) = \text{argmin}_{q(\mathbf{z})\in\mathcal{Q}}\text{KL}(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})). \tag{2}$$

Much recent effort has gone into solving VI problems in different settings (Bouchard-Côté and Jordan, 2010; Ingraham and Marks, 2017; Zhang *et al.*, 2019). VI can be framed as solving an optimization problem over function families: neural networks are popular candidates for representing and learning complex functions. VI was incorporated into autoencoders (Kingma and Welling, 2014) to create the architecture of a variational autoencoder (VAE). A VAE consists of an encoder network to approximate $p(\mathbf{z}|\mathbf{x})$ through a function $q_{\mathbf{x}}(\mathbf{z})$, and a decoder network $p(\mathbf{x}|\mathbf{z})$(Fig. 1A). Conceptually, the encoder solves an inference problem: approximating the posterior distribution $p(\mathbf{z}|\mathbf{x})$ as some $q^*_{\mathbf{x}}(\mathbf{z})$, while the decoder solves a reconstruction problem: defining a generative process for $p(\mathbf{x}|\mathbf{z})$, given the latent variables. The VAE posterior is modeled by a multivariate normal $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ of the same dimension as $\mathbf{z}$. Training then comes down to minimizing two objective functions. For the encoder network, which should learn a 'well distributed' latent space, minimize the KL divergence: $\text{KL}(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})||\mathcal{N}(0, \mathbf{I}))$. For the decoder network, which should reconstruct the inputs $\mathbf{x}$ from the latent space, minimizing either an $L1$ or $L2$ objective function with respect to $\hat{\mathbf{x}}$ is appropriate. The use of KL-divergence and an $L2$ objective solves the VI formulation of Equation 2 (Kingma and Welling, 2014), however, an $L1$ objective may be preferred in practice, e.g. in cases where we want to suppress the effects of outliers on the structure of $\mathbf{z}$ (Botchkarev, 2018).

2.2 RVAgene: a recurrent variational autoencoder to model gene expression dynamics

Following the VAE architecture, RVAgene consists of an encoder and a decoder network with a reparameterization step in between. To incorporate the knowledge that we are modeling temporal data, recurrent neural networks offer an ideal architecture to use for both the encoder and the decoder networks. Recurrent and VAE networks have been successfully combined elsewhere, e.g. for textual (Nallapati *et al.*, 2016) and time series data (Malhotra *et al.*, 2015).

The architecture of RVAgene is based on Fabius and van Amersfoort (2014). An input sequence (i.e. gene) $x \in \mathbf{x}$, $x = (x_1, x_2, \ldots, x_t, \ldots, x_T)$ is encoded using a recurrent function described by a long short-term memory (LSTM) unit. LSTM units are the state-of-the-art in recurrent architectures, since they are robust against the vanishing gradient problem for longer sequences, unlike other recurrent units (see details in Hochreiter and Schmidhuber (1997)). We encode x in the following manner:

$$h^{\text{enc}}_{t+1} = \text{LSTM}(W^T_{\text{enc}}h^{\text{enc}}_t + W^T_{\text{inp}}x_t + b_{\text{enc}}), \tag{3}$$

where (W_{enc}, W_{inp} and b_{enc}) are network weight parameters, and the hidden states h_t represent information shared over timepoints in the LSTM. The dimension of the h_t (and W_{enc}) is given by a hyperparameter ('hidden-size'). The encoded h_{t+1} are used to parametrize the posterior mean and variance from x, with mean $_{\mu z}$ and diagonal covariance $_{\sigma z}$ as:

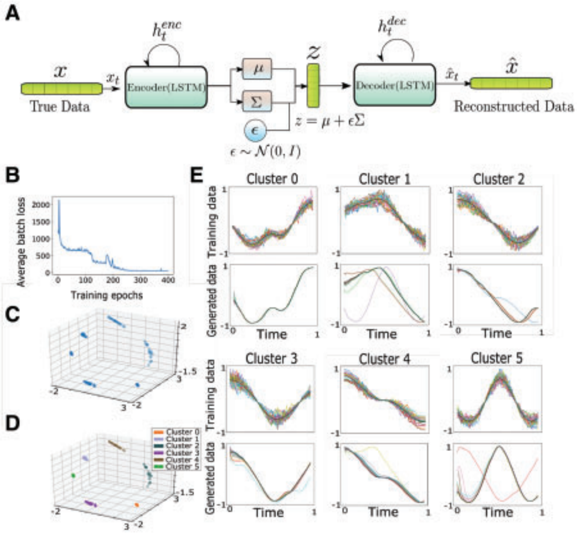

Fig. 1. Unsupervised representation learning with RVAgene using synthetic data. (**A**) Schematic diagram of the RVAgene model. (**B**) Average loss function ℓ as over duration of training. (**C**) Latent space representation learnt by RVAgene model after training. (**D**) Clusters detected by *k*-means clustering on the latent space, with $k = 6$ (**E**) First and third rows show input training data used (20 simulated genes in each of six clusters); cluster means shown in black. Second and fourth rows show the model-generated data, obtained by sampling and decoding points from the latent space; decoded cluster empirical means shown in black

$$\mu_z = W_\mu^T h_{T+1}^{\text{enc}} + b_\mu$$

$$log(\sigma_z) = W_\sigma^T h_{T+1}^{\text{enc}} + b_\sigma. \tag{4}$$

We then use the reparameterization step described in Kingma and Welling (2014) to sample z from the distribution:

$$z = \mu_z + \epsilon\sigma_z, \tag{5}$$

where, for known $_\epsilon$, backpropagation through the sampling step is possible while training the network.

For the decoder network, the first state h_1 is calculated from z, and the recurrent formulation follows by reconstructing x as $\hat{x} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_t, \ldots, \hat{x}_T)$, thus:

$$h_1^{\text{dec}} = \text{sigm}(W_z^T z + b_z)$$

$$h_{t+1}^{\text{dec}} = \text{LSTM}(W_{\text{dec}}^T h_t^{\text{dec}} + W_{\text{out}}^T \hat{x}_t + b_{\text{dec}}) \tag{6}$$

$$\hat{x}_t = \text{sigm}(W_{\text{out}}^T h_t^{\text{dec}} + b_{\text{out}}),$$

where $\text{sigm}(u) = \frac{1}{1+e^{-u}}$ is the sigmoid activation function, and (W_i, b_i) are the network weight parameters. A schematic diagram of the network is shown in Figure 1A, which can now be trained using backpropagation, to minimize the objective function:

$$\ell(\theta, x) = D_{KL}(\mathcal{N}(\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z) || \mathcal{N}(0, \mathbf{I})) + |x - \hat{x}|, \tag{7}$$

where $\boldsymbol{\mu}_z$ and $\boldsymbol{\Sigma}_z = \text{diag}(\sigma_z)$ are calculated from x by the encoder.

To evaluate the accuracy of RVAgene, we need an appropriate error measure. For each gene in the test set, we calculate the $L1$ reconstruction error between generated data $\hat{x}$ and true data x, averaged over all time points. We normalize the data to lie in $[0, 1]$ to avoid skewing the error by differences in gene expression magnitudes. Thus we define:

$$\text{Reconstruction error}(x, \hat{x}) = \frac{1}{T} \sum_t |s(\hat{x})_t - s(x)_t|, \tag{8}$$

$$\text{where, } s(x) = \frac{x}{\sum_{t=1}^{T} x_t}. \tag{9}$$

2.3 Generating synthetic gene expression time series data

To test RVAgene, we generate a synthetic time series dataset. Six clusters each containing 20 genes are simulated, where for each cluster c, the mean gene expression time series $Y_c = (y_{c1}, y_{c2}, \ldots, y_{ct})$ was generated using addition or convolution and rescaling of two random sinusoidal functions of the form $k_1 \sin(k_2 t)$, where k_1, k_2 are randomly chosen positive integers. Trajectories of cluster members were then generated by sampling from the multivariate normal $\mathcal{N}(Y_c, \Sigma_c)$. We model $\boldsymbol{\Sigma}c$ as the positive definite matrix $\alpha Y_c Y_c^T$, where $_\alpha$ is a scaling factor, we use: $\alpha = 1/|Y_c Y_c^T|$. As defined, $\boldsymbol{\Sigma}c$ will describe non-zero correlations for all pairs of time points, (t_i, t_j). This is unrealistic, so we set to 0 the entries of $\boldsymbol{\Sigma}c$ for which column and row indices have a difference of more than some threshold T (we used $T = 50$), reflecting the fact that correlations between time points are lost over larger time windows (temporal correlations are local). Note that under this condition, $\boldsymbol{\Sigma}c$ is no longer necessarily positive definite. The multivariate Gaussian sampler numpy.random.multivariate_normal() implemented in numpy (Harris *et al.*, 2020) was used to sample from this augmented $\boldsymbol{\Sigma}c$. After generating a simulated dataset by this process, we also added Gaussian noise, drawn from $\mathcal{N}(0, 0.7)$, to the simulated dataset to produce an additional dataset exhibiting higher levels of noise.

3 Results

3.1 RVAgene can accurately and efficiently reconstruct temporal profiles from synthetic data

We generated a dataset of 120 genes using convolutions of sinusoidal functions (see Section 2) to test the ability of RVAgene (Fig. 1A) to learn and reconstruct noisy non-linear temporal profiles. An RVAgene model was trained on all 120 genes from 6 clusters with a hidden size of 70 and a 3 dimensional latent space. The model was trained for 400 epochs, after which the average batch objective ℓ function indicates convergence (Fig. 1B), producing a three-dimensional latent space representation (Fig. 1C). K-means clustering on the latent space ($k = 6$) identified well-separated clusters (Fig. 1D).

RVAgene modeling followed by k-means clustering on the latent space identified six clusters with perfect fidelity between predicted and true clusters. One might reasonably ask, why use a neural network for this task? Simpler dimensionality reduction methods (e.g.

PCA, t-SNE or a non-variational autoencoder) would also find the correct solution. RVAgene has the advantage over these methods that the underlying structure of the latent space leads to interpretability. A point in reduced PCA or t-SNE space that does not overlap with a data point is not interpretable. Traditional autoencoders lack regularity in the latent space, i.e. even for a representation with arbitrary accuracy (a reconstruction error of zero), decoding a point that does not correspond to a training data point can result in non-sensical generated data, even if the decoded point is arbitrarily close to a training data point. Variational Auoencoders remedy this by learning a regularized or smoother distribution on the latent space. In this sense, the KL-divergence term in the VAE loss function can be thought of as a regularizer. This property enables RVAgene to generate new gene expression dynamics by decoding points from different regions of the latent space, having properties similar to clusters nearby to those points.

To demonstrate the generative properties of the RVAgene latent space, we sample points from multivariate Normal distributions, centered on the empirical mean of each cluster with variance of 0.4, i.e. $\mathcal{N}(\mu_c, 0.4I)$, where $_{\mu c}$ is the empirical mean of the cluster and **I** is the identity matrix in $\mathbb{R}^3$. Corresponding to each cluster, we sample 20 points in the latent space, and use the decoder network to generate new time series data (Fig. 1E). Most of the points sampled generate trajectories that belong to the correct cluster. Moreover, we identify cases corresponding to transitions between clusters. For example, some points sampled near Cluster 2 generate trajectories that are similar to members of Cluster 4, and vice versa. This makes sense due to the similarity between the temporal profiles of Clusters 2 and 4. A similar correspondence is observed between Clusters 1 and 5. We note that in a few cases the generated data have profiles that differ from their cluster of origin and appear most similar to those of another cluster. This occurs when points are sampled close to neighboring clusters, e.g. the red line for Cluster 5 in Fig. 1E has been sampled from a point close to cluster 3. We also observe some generated trajectories that display intermediate profiles between two or more clusters: the decoder function learnt by RVAgene is smooth, and gives rise to meaningful representations of points across regions of the latent space.

RVAgene offers additional functionality as a tool for removing noise from the data. Via sampling and decoding points from the latent space, RVAgene reconstructs trajectories that are smooth and de-noised relative to the input data (Fig. 1E, Supplementary Fig. S1). Similar neural network approaches have been proposed to denoise from single-cell data, e.g. using a deep count autoencoder (Eraslan *et al.*, 2019). RVAgene provides data denoising as a by-product of its primary functionality: learning patterns of dynamic gene expression.

To investigate the impact of input noise levels on RVAgene performance, we added Gaussian noise drawn from $\mathcal{N}(0, 0.7)$ to the simulated data to produce a dataset with higher overall noise levels. RVAgene learns a latent space shown in (Supplementary Fig. S1A) from which six clusters are identified by k-means clustering (Supplementary Fig. S1B). It is notable that the clusters identified in the latent space are not as clear in this case as for lower noise levels (Fig. 1), however RVAgene can still reconstruct the distinct profiles with high confidence. To illustrate this, we plot the original training data alongside model-generated data, sampled at random points in the latent space from $\mathcal{N}(\mu, 0.4\mathbb{I})$ around each cluster mean $_{\mu}$ for each of the six clusters (Supplementary Fig. S1C). From these simulations, RVAgene appears able to separate even relatively high levels of noise from the signal, in order to learn a smooth encoding and corresponding generative process for distinct temporal patterns.

It is inevitably challenging to include sufficient dimensionality and variation in synthetic datasets to accurately capture biological processes such as those we observe in experimental datasets. Thus, in the subsequent two sections, we test the capabilities of RVAgene on two whole-genome biological datasets: embryonic stem cell differentiation, and kidney injury response. As we will see, in these cases it may not be possible to characterize the latent space by simple (e.g. k-means) clustering; we need to use other means to gain insight into the features of the latent space.

3.2 RVAgene modeling of pseudotemporally ordered data during embryonic stem cell differentiation

We applied RVAgene to model gene expression dynamics during embryonic stem cell (ESC) differentiation. Klein *et al.* (2015) identified 732 differentially expressed genes over the time course of mouse ESC differentiation following leukemia inhibitory factor (LIF) withdrawal. Data is gathered at four time points: 0, 2, 4 and 7 days after LIF withdrawal (Supplementary Table S2 in Klein *et al.*, 2015). We ordered the data (2717 single cells) using diffusion pseudotime (DPT), which provides robust methods for the reconstruction of single-cell temporal processes (Haghverdi *et al.*, 2016). The root cell was randomly sampled from the initial time point (Fig. 2A). The inferred pseudotime is highly correlated with the experimental time points, giving confidence that true biological processes are represented over the DPT pseudotime. The gene expression dynamics over pseudotime show considerable variability among cells. To smooth the data, we apply a moving window average, over windows of length 40, to give 68 time points after smoothing (Fig. 2A). We fit linear regression models to the smoothed pseudotime profiles of each gene (Supplementary Fig. S2), and see that for the majority of genes the correlation coefficients are >0.5 (Fig. 2B), with a clear distinction between the up- and down-regulated genes over pseudotime.

An RVAgene model was trained on the data with a two-dimensional latent space, on which genes are classified based on their correlation coefficients (Fig. 2C). Two distinctive characteristics emerge: (i) the two groups (up- and down-regulated genes) are well-separated in the latent space, and (ii) the two groups merge and overlap at some point, illustrating the continuity of the latent space, as discussed above. We compared the results of RVAgene with DPGP, an unsupervised approach for gene expression time series clustering (McDowell *et al.*, 2018). DPGP is a hierarchical Bayesian model that estimates the number of clusters along with the cluster membership.

To assess the correspondence between methods, genes clustered by DPGP (Supplementary Fig. S3) were projected onto the RVAgene latent space (Fig. 2D). Of the 12 clusters detected by DPGP, the four largest can be characterized by their up- and down-regulation profiles over pseudotime. On the RVAgene latent space, we find that genes sampled from each of the DPGP clusters appear close together, and moreover, are represented on a spectrum from upregulation to downregulation (Fig. 2D). The goals of RVAgene and DPGP are to some degree complementary: DPGP characterizes gene expression profiles discretely with no need for prior information, while RVAgene characterizes profiles with a continuous representation, that can explain smooth changes in patterns. To assess the ability of the model to reconstruct genes not used during training, we kept aside 300 genes for testing and trained RVAgene on the remaining 432 genes. We note that in this case (and in the case of single-cell datasets in general), the generative model of RVAgene produces pseudotime-smoothed gene expression trajectories, rather than being generative of raw pseudotemporal data, which tend to display overall high noise levels. Reconstructed test gene expression profiles are shown for three reconstructed genes (Fig. 2E), chosen to sample across the spectrum of reconstruction errors (Fig. 2F). The reconstruction for *Ddt*, which has a reconstruction error near the mode (Fig. 2F), shows very high accuracy. The reconstruction for *Hmgb2*, which has twice the reconstruction error, still broadly captures the temporal profile but with lesser accuracy. Finally we show the reconstruction for *Rhox4e*, a gene that was sampled from the long tail of the reconstruction error distribution, i.e. does not well match the data. Comparing these three examples with the full distribution of reconstruction errors (Fig. 2F), we see that the large majority of genes lie to the left of *Hmgb2*, i.e. have better-than-moderate accuracy. The reconstruction error of *Hmgb2* is close to 0.005, which we use as a cut off for 'well-reconstructed' genes, based on analysis of individual gene reconstructions. The cumulative reconstruction error distribution reiterates this point: 230 out of 300 genes (77%) have a reconstruction error $\leq$ 0.005 (Fig. 2G); we can conclude that the majority of test genes were faithfully reconstructed by the model.

RVAgene accurately reconstructed most gene profiles using only ~60% of the data for training (Fig. 2G), likely due to co-regulation of gene expression programs. This led to a question:

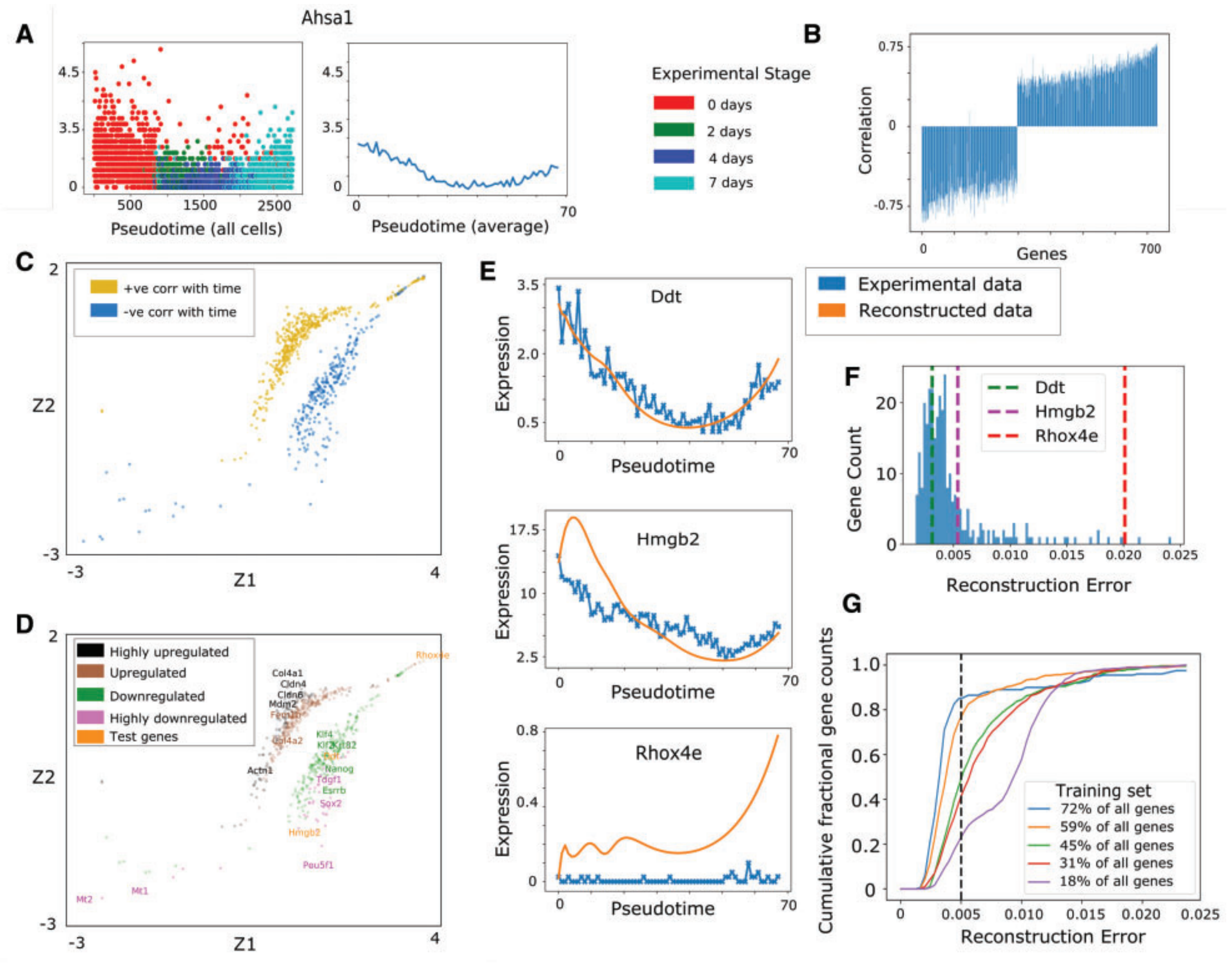

Fig. 2. Accurate reconstruction of embryonic stem cell differentiation dynamics with RVAgene. (**A**) Pseudotemporal ordering of 2717 single cells (data from Klein *et al.*, 2015), calculated using DPT; example gene shown: Ahsa1. Gene expression values given as log2(counts + 1) for all cells (left), and for sliding window average (right). (**B**) Pearson correlation coefficient between gene expression and time for 732 differentially expressed genes. (**C**) The 2 D latent space learnt by an RVAgene model trained on 732 gene profiles over pseudotime, showing clear separation between upregulated and downregulated genes. (**D**) Comparison of RVAgene and DPGP. The four largest clusters from DPGP are plotted on the RVAgene latent space: temporal expression patterns (from highly upregulated to highly downregulated) are in close agreement between methods. (**E**) Comparison of experimental data and reconstructions. Model-generated reconstructions of three genes from the test set not used in training: Ddt, Hmgb2 and Rhox4e. Expression values are log2(counts + 1). (**F**) Distribution of average $L1$ reconstruction errors for the 300 genes used in the test set. Genes plotted in C are marked. (**G**) Cumulative distributions of reconstruction errors on randomly sampled sets of test genes, where the full data were split into test groups of: 200 genes (train on 72%), 300 genes (train on 59%), 400 genes (train on 45%), 500 genes (train on 31%) and 600 genes (train on 18%)

what is the smallest training gene set that can be used to accurately reconstruct gene dynamics? We subset the data randomly into train/test sets and trained separate RVAgene models on each. We found that reconstruction errors slowly increase as the size of the training set decreases, but not until the training set was as low as 18% of the data did the reconstruction errors significantly increase (Fig. 2G, Supplementary Fig. S4). Analysis of the cumulative distribution of reconstruction errors across all groups found that RVAgene reconstructs the majority of gene temporal profiles well (defined as below a reconstruction error of 0.005) if $\geq$ 45% of the data is used for training. The successful reconstruction of gene expression dynamics de novo while training on small subsets of the data suggests widespread co-regulation of gene expression programs during embryonic stem cell differentiation, as found in previous work (Jang *et al.*, 2017).

3.3 Comparison of RVAgene with alternative approaches for gene clustering

In order to assess the performance of RVAgene for gene clustering and biological discovery, we compared it to five alternative methods: two neural network approaches and three hierarchical clustering methods. To assess the utility of the recurrent architecture of RVAgene, we trained non-recurrent (i.e. fully connected) variational autoencoders on the embryonic stem cell differentiation dataset (Klein *et al.*, 2015). We compared two options: using the pseudotemporally ordered and smoothed data as input (same as for RVAgene), or using the raw (i.e. unordered and unsmoothed) gene expression data as input. We trained encoder and decoder networks of depth two (one hidden layer) and with a hidden layer size of 400 (we performed a hyperparameter search to optimize this). Theoretically, depth two networks are large enough to learn any non-linear function (Barron, 1994; Cybenko, 1989; Funahashi, 1989; Hornik *et al.*, 1989), although the fully connected VAE has no recurrent inductive bias. Thus we test how important this recurrent inductive bias is in practice.

The results of the comparison of neural networks are given in Fig. 3A and B. In each case, models were trained for 200 epochs. Annotating the results in latent space using correlations against pseudotime (Fig. 3A) shows that all three models separate the data reasonably well, with slightly better separation for the recurrent architecture (RVAgene). We also annotated the results using cluster labels from the largest four DPGP clusters for comparison. These are appropriate 'gold-standard' cluster labels since robust dynamical signatures are learnt by DPGP in each case (Supplementary Fig. S3). RVAgene captures: (i) better separation between clusters that either of the non-recurrent networks, and (ii) a spectrum of behaviors from up- to down-regulated (Fig. 3B).

We also performed hierarchical clustering on the pseudotemporally ordered and smoothed data using three standard hierarchical clustering methods: the Nearest Point Algorithm, the Farthest Point Algorithm and UPGMA (the Unweighted Pair Group Method with Arithmetic mean). We annotated the results with the same clusters labels from DPGP (Fig. 3C). UPGMA performs best out of these three clustering algorithms, yet still does not attain clear seperation between each of the four groups. Thus, the 2 D latent space representation of RVAgene is better than both 1 D representations via hierarchical clustering and the alternative neural network latent space representations at distinguishing between dynamic gene profiles in pseudotemporally ordered data.

3.4 RVAgene can classify and predict gene expression dynamics in response to kidney injury

We investigated gene expression dynamics in the murine kidney by applying RVAgene to a dataset that describes gene expression profiles before, during and after a kidney injury (Liu *et al.*, 2017). The

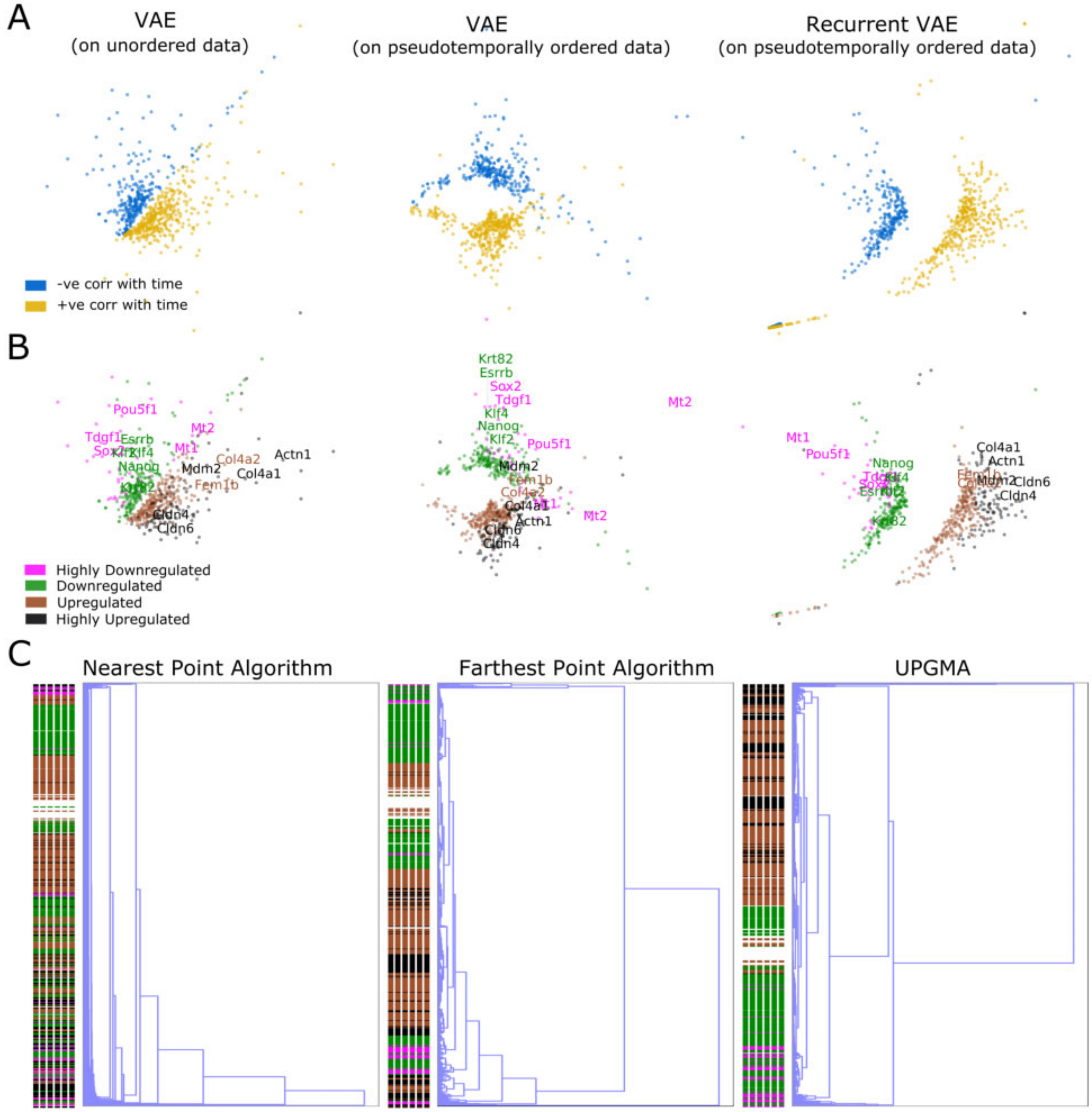

Fig. 3. Comparison of information captured in RVAgene latent space compared to a standard fully connected VAE and results of standard hierarchical clusterings. (**A**) Here, we show latent spaces learned by fully connected VAE and RVAgene. The pseudotemporally ordered data was also smoothed. (**B**) We annotate the learned latent spaces using the top 4 clusters detected by DPGP on this dataset. In all three of these cases we report best results after relevant hyperparameter search and optimal training. (**C**) We perform standard hierarchical clusterings (Nearest Point Algorithm, Farthest Point Algorithm and UPGMA (Unweighted Pair Group Method with Arithmetic mean)) on pseudotemporally ordered and smoothed ESC data and annotate the learned representation in the same manner as in (B)

dataset is temporally rich, with a total of ten bulk samples over twelve months. Since in this case no single-cell information is available, we cannot order samples by pseudotime to smooth the data. Moreover, the temporal gene expression profiles described in Liu *et al.* (2017) display more complex dynamics than for the previous dataset (Klein *et al.*, 2015), and are not readily separable by linear patterns of up- and down-regulated genes (cf. Fig. 2C). Thus, below, we must consider non-linear models in order to characterize the temporal patterns observed.

The data consist of one initial timepoint ($t = 0$) before the injury event (an ischemia/reperfusion injury model) and nine subsequent time points ($t = 1$ to 10) following the injury (48, 72 h, 7 days, 14 days, 28 days, 6 months and 12 months). We note that the timepoints are not uniformly spaced, which is not taken into account in RVAgene, which only models the broad temporal trend (see Discussion). From an initial list of 1927 differentially expressed genes measured over the time course in three biological replicates, we removed putative/predicted and non-protein coding genes, retaining a list of 1713 genes as input to the model.

We ran RVAgene separately for each of three biological replicates. Independent replicates and independently trained models provide additional means with which to test the reproducibility of these methods. For each replicate, RVAgene was trained with a two-dimensional latent space and a hidden size of 10, on the full set of genes over 200 epochs: found to be sufficient for the convergence of ℓ (see Section 2 for further details). We fit linear regression models to the temporal gene profiles (Supplementary Fig. S5) and found that linear fits did not describe the gene temporal profiles well (most correlation coefficients had values close to zero),nor they did not identify separate clusters in the latent space. Normalizing the data to lie in $[0, 1]$ improved our ability to discriminate clusters in the latent space (Supplementary Fig. S5C), but came at the expense of a significant loss of information, as the variance captured in the latent space was dramatically reduced. The absence of evidence for linear correlations could indicate expression dynamics that are uncorrelated with time, but could of course also indicate more complicated (non-linear) gene expression dynamics, which are explored below.

To study non-linear gene expression dynamics, we fit a 2nd degree polynomial, i.e. we fit the temporal trajectory of each gene x to: $x = at^2 + bt + c$, where a, b, c are constants (Supplementary Fig. S6). We hypothesized that this function could adequately describe the transient dynamics observed by Liu *et al.* (2017) for most genes in response to the kidney injury. Thus, we classified genes into one of two groups, $a < 0$: convex (up-down pattern), 1200 genes; and $a \geq 0$: concave (down-up pattern), 512 genes. In the latent space, the separation of these two groups is clearly visible for each replicate (Fig. 4A). Moreover, the classification is in agreement with Liu *et al.* (2017), where the majority of differentially expressed genes are upregulated transiently. To explore the ability of RVAgene to reconstruct gene expression profiles not used in model development, we kept aside 300 randomly sampled genes for testing, and trained RVAgene models on the remaining genes for each of the three replicates. Independently for each model, we then generated dynamic profiles for the test genes. Three genes sampled randomly from the test set are plotted in Fig. 4B. Of particular note, for each of genes, the model-generated data captures the temporal patterns while displaying a higher degree of similarity across replicates than the experimental data itself. This illustrates that the model is neither under- nor overfitting, but capturing the underlying biological patterns while sufficiently accounting for the noise. Reconstruction errors are comparable across the three replicates, albeit with slightly higher overall errors in replicate 1 (Fig. 4C and D). Overall, the reconstruction errors are higher than for the previous section (averaging over many pseudotemporal time points allowed us to significantly reduced the noise). To investigate in more depth the features that are captured in the RVAgene latent space, we performed two sets of analyses: unbiased clustering, and targeted exploration. For the unbiased analysis, we performed k-means clustering on RVAgene latent space of replicate 1 (R1) with $k = 9$ (Supplementary Fig. S7A); we project the clusters labels learnt onto replicates R2 and R3. All cluster identities are well-preserved across replicates, with the exception of cluster 5, which seems to indicate outlier genes in R1. To study biological processes within these clusters, we performed GO term enrichment analysis on each. In Supplementary Figure S7B we plot one significant GO term per cluster (omitting cluster 5), and see that specific regions of the latent spaces across replicates can be characterized in terms of biological processes, many of which relate to metabolic and immune system responses. These can be separated into two broad classes, which separate the left-hand side of R1 (metabolic processes downregulated during injury response) from the right-hand side (immune responses upregulated during injury response).

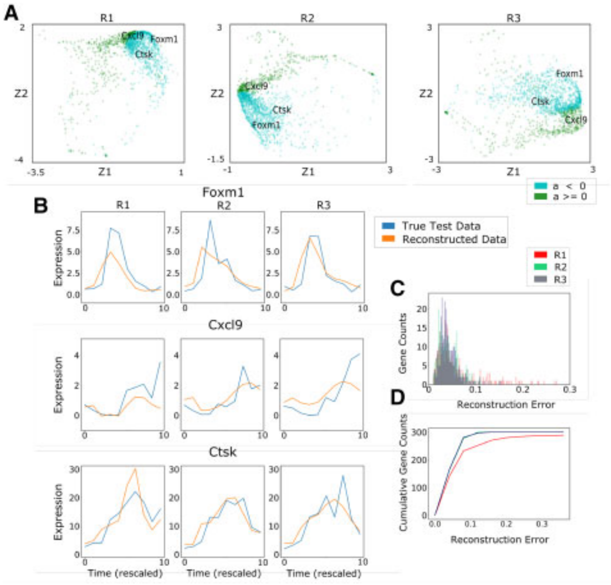

Fig. 4. Accurate reconstruction of kidney injury response gene dynamics with RVAgene. (**A**) Latent space representations of RVAgene models trained separately on three independent replicates (R1–R3); classified by quadratic fit coefficient *a*. (**B**) Model generation of gene dynamics for genes not used in training: *Foxm1, Cxcl9* and *Ctsk*. (**C**) Histograms of reconstruction errors for RVAgene models trained on R1–R3 (truncated). (**D**) Cumulative distribution of reconstruction errors

To study the effects of gene-specific regions of the latent space in greater depth, we chose three distinct regions based on the co-location of genes of interest. These gene groups studied on the latent space are: (i) a *Wnt* group consisting of family members *Wnt2* and *Wnt4*; (ii) an *Slc* group consisting of family members *Slc7a13* and *Slc22a18*; and (iii) a *Sdc1* group, consisting of only *Sdc1*. For each group, we characterized neighboring genes by defining a circular neighborhood around each gene in the group, with radius r (depending on the local density, the radius was varied, giving: $r^2 = 1$ for *Slc*, $r^2 = 0.3$ for *Sdc*, $r^2 = 0.05$ for *Wnt*). We then took all genes inside this radius for each replicate, and found the intersection of genes over the three replicates (Fig. 5A and B). We analyzed the intersection gene set for each group by studying their temporal profiles and their gene ontology (GO) term associations. Each group was characterized by a strikingly clear temporal profile. The *Sdc1* and *Wnt* groups both show transient upregulation, over different timescales: the *Sdc1* group is upregulated from 24 h post-injury until 14–28 days post-injury (fast response) (Supplementary Fig. S8B), whereas the *Wnt* group is upregulated at 7 days post-injury until 28 days post-injury (slow response) (Fig. 5C). In contrast, the *Slc* group is downregulated at 24 h post-injury, and remains suppressed until 7–28 days post-injury (Fig. 5D).

Analysis of GO biological process terms enriched in each gene group further highlighted the power of the latent space for biological discovery. The fast response (*Sdc1*) group was characterized by upregulation of programs related to apoptosis, stress response, wound healing and chemotaxis, i.e. the first responders to the site of injury (Supplementary Fig. S8C). In addition all five *Lox* genes comprising the GO term ‘peptidyl-lysine oxidization’ were found in this group. This is consistent with the oxidative stress resulting from the renal ischemia-reperfusion injury that was performed. However,

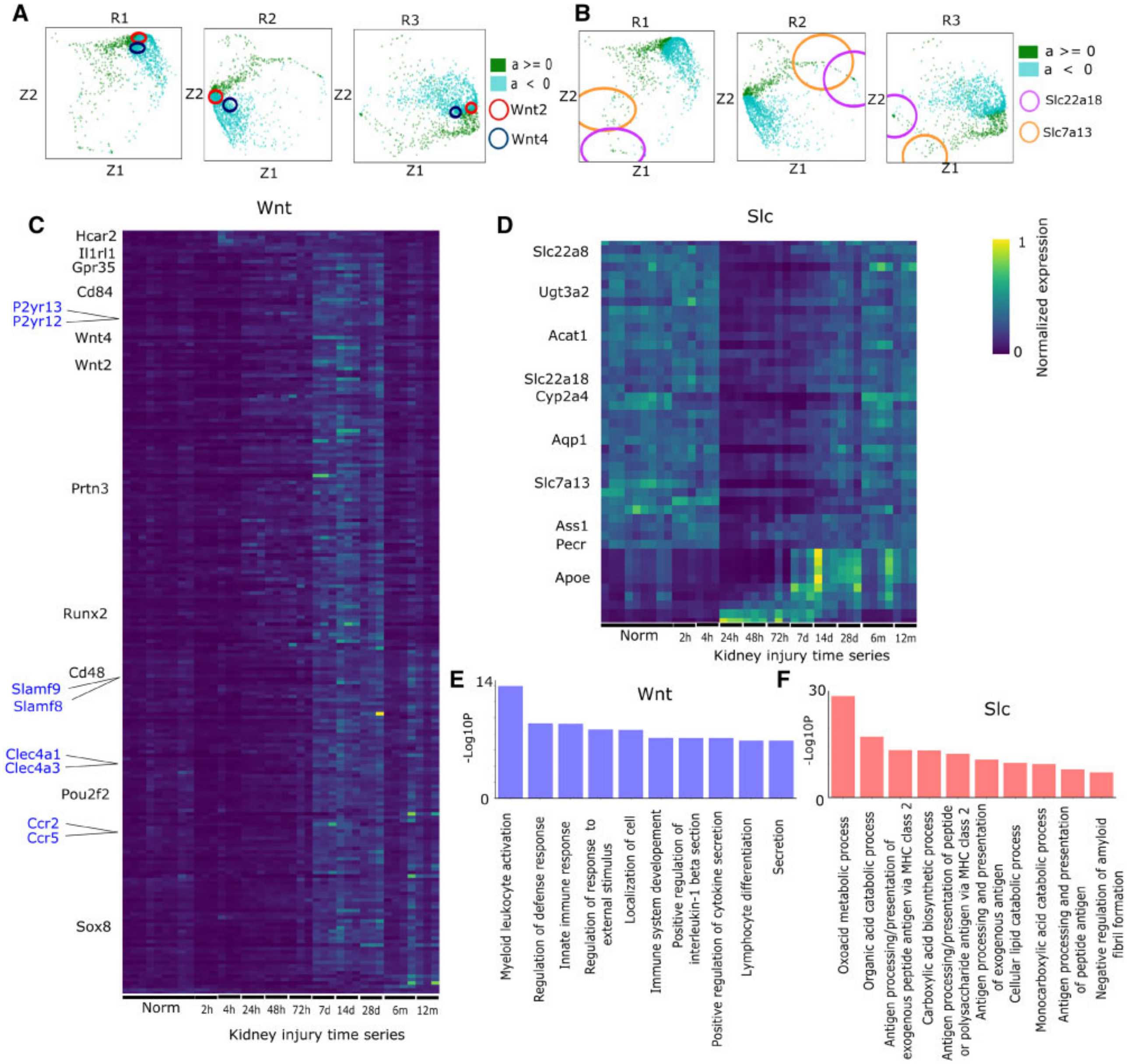

Fig. 5. RVAgene latent space captures biological processes driving concordant gene expression changes. (**A**) Z-plots for replicates R1–R3 with local neighborhoods of Wnt2 and Wnt4 marked (circles). (**B**) As in A, for Slc family members Slc22a18 and Slc7a13. (**C**) Heatmap of expression changes over time course of injury for the Wnt neighborhood genes in the intersection of R1-R3. Selected genes marked (black), as well as ortholog gene pairs (blue). (**D**) As in C, for Slc neighborhood genes. (**E**) Histogram of -log10 *P* values of gene ontology terms for biological processes terms associated with the Wnt neighborhood (gene set in C). (**F**) As in E, with the Slc neighborhood (gene set in D)

distinct factors regulate the *Lox* family genes, as can be partly observed by their subtle differences in temporal profile (Supplementary Fig. S8D). Their co-location in the latent spaces of all three models thus highlights the potential use of RVAgene for discovery of complex temporal regulatory events from gene expression data.

The slow response (*Wnt*) group was primarily characterized by immune response processes, including leukocyte activation, platelet aggregation and various cytokine-mediated pathways including *IL-1* and *IL-33* (Fig. 5E). Notably, the Wnt group identies multiple gene orthologs (Fig. 5C) with very similar profiles: likely evidence of shared temporal regulation. This illustrates once again (as for the *Lox* genes above) the potency of RVAgene for the discovery of temporally co-regulated genes.

Finally, the *Slc* group of genes shows a transiently down-regulated pattern between 24 h and 7–28 days, although some gene in this group deviate from this pattern (Fig. 5D). GO term enrichment identifies the positive regulation of metabolic processes (Fig. 5F). The downregulation of metabolic programs during the response to kidney injury is agreement with the findings of Liu *et al.* (2017). Notably, this metabolism-sensitive group contains many genes that also display sexually dimorphic expression, primarily in specific regions of the proximal tubule (Ransick *et al.*, 2019), thus independently identifying the well-established (though under-studied) interplay between sex differences and injury responses in the kidney (Neugarten *et al.*, 2000).

In summary, unsupervised analysis of groups of genes co-located in the latent spaces of RVAgene finds: (i) high similarity between temporal gene profiles of genes nearby in latent space, and (ii) clear biological signatures represented by these groups of nearby genes, in strong agreement with prior knowledge (Liu *et al.*, 2017). Moreover, the latent spaces of RVAgene models can be used to predict programs of temporal co-regulation.

3.5 Assessment of the computational efficiency of RVAgene

We assessed the computational efficiency of RVAgene for various settings and hardware. For the majority of the models trained, 100–200 epochs was sufficient for the loss function ℓ to converge. For tests performed here, we recorded the RVAgene runtime for 100 epochs of training on Klein *et al.* (2015), using models that varied in their number of genes and time points. In each case we used a latent space of dimension two, a hidden size of 10 and a training batch size of 10. We ran the model on an intel i7 CPU with four cores and a Tesla K20 GPU. Runtimes were recorded on linux via the inbuilt time script (/usr/bin/time –verbose). As the number of time points and genes grew large (up to 60 time points and 700 genes), total runtimes on CPU were on the order of 10^3s (<20 min) (Fig. 6A). On GPU, total runtimes were decreased to around 100 s (<3 min). Thus, RVAgene is readily scalable to tens of thousands of genes and hundreds of time points for training times of up to a few days on CPU or hours on GPU. For comparison, as described in McDowell *et al.* (2018), the approximation-free time complexity of each iteration of learning for DPGP is $\mathcal{O}(GT^3)$, due to the G matrix inversions, each of size $T \times T$, for a dataset with G genes and T timepoints. The complexity for each epoch of training of RVAgene is $\mathcal{O}(GT)$.

In terms of peak memory usage, since RVAgene is a neural network trained using backpropagation (Rumelhart *et al.*, 1986), maximum memory used during training is of the same size as the network itself, which is constant given that the model parameters are fixed (Fig. 6B). This is in contrast to Gaussian Processes (such as DPGP), which initially assign each gene to its own cluster, thus must store G matrices of size $T \times T$, for G genes and T timepoints per gene. This leads to quickly increasing runtime peak resident set sizes for DPGP compared to RVAgene (Fig. 6B Inset). The memory used by DPGP grows with the number of time points as $\mathcal{O}(GT^2)$). Thus, DPGP will not run with large numbers of genes and time points. A note on this comparison: it is not direct, in the sense that DPGP performs clustering and RVAgene does not, in addition to other important differences between the goals of the methods. Nonetheless, the size and scope of current biological datasets—particularly at single-cell resolution—in many cases preclude the use of DPGP without large reductions of the input data size. As we have shown, a feasible and efficient alternative in such cases is to run RVAgene, and then to perform clustering or other classification analyses post hoc on the latent space of the model.

4 Discussion

We have presented RVAgene, a recurrent variational autoencoder for generative modeling of gene expression time series data. Through its encoder network, RVAgene provides means to visualize and classify gene expression dynamic profiles, which can lead to the discovery of biological processes. Through its decoder network, RVAgene provides means to generate new gene expression dynamic profiles of either the full data or (in the case of single-cell studies) the pseudotime-smoothed data by sampling points from the latent space. In doing so, RVAgene can accurately reconstruct gene dynamics in complex biological data. As a by-product, on single-cell datasets the model directly produces smoothed outputs, useful for denoising gene expression time series data. RVAgene is efficient on temporally rich whole genome datasets, in comparison to current existing methods.

RVAgene can be used to discover structure in the data, such as gene profile clusters. Popular methods for clustering gene profiles such as Bayesian hierarchical clustering (Cooke *et al.*, 2011) or DPGP (McDowell *et al.*, 2018) detect the number of clusters in the data by fitting a hyperparameter α, the concentration parameter of the governing Dirichlet process (Ferguson, 1973). Although unsupervised, inevitably, the choice of α affects the number of clusters output. Visualizing the data first with RVAgene can give an idea whether the data favor clustering or a continuous representation. Thus analysis in RVAgene can guide the setting of the hyperparameter α in DPGP and similar methods. In the case of ESC differentiation, DPGP predicts 12 clusters (Supplementary Fig. S3), yet most have very few members and many share similar patterns. The RVAgene latent space for this dataset finds two major divisions in the data, and orders the largest DPGP clusters along a spectrum (Fig. 2D), suggesting that DPGP might be overfitting the data. Indeed, the two methods can be used complementarily: RVAgene for high-level structure discovery and DPGP for clustering. In cases where learning a detailed noise model (at single time point resolution) is important to the user, DPGP or other Gaussian Process models are preferable over RVAgene. However, DPGP does not scale well with large datasets and thus cannot always be used (Fig. 6).

The latent space of an RVAgene model encodes useful information about biological features, and in that sense provides biologically interpretable representations of the data. However, the representation is not interpretable in the sense that the components of the latent space do not have a physical meaning nor are they necessarily independent. Recent methods have tackled this issue of interpretability, by either modifying the loss function to make components independent (Higgins *et al.*, 2016) or substituting linear functions in parts of the VAE (Ainsworth *et al.*, 2018; Svensson *et al.*, 2020). These methods have clear advantages regarding the analysis and interpretation of features in the latent space. In future work, decoding an RVAgene model with a linear function (Svensson *et al.*, 2020) could facilitate additional discovery and improve our ability to gain insight into dynamic biological processes through the analysis of the latent space.

Dynamic changes in gene expression underlie essential cell processes. As such, modeling gene expression changes can also facilitate downstream analysis tasks, including gene regulatory network (GRN) inference. Inferring gene regulatory networks from single-cell data is challenging (Chen and Mar, 2018), particularly due to cell–cell heterogeneity and high levels of noise. Several recent approaches to GRN inference make use of temporal profiles (Deshpande *et al.*, 2019; Kim *et al.*, 2021) or differential equations (Aubin-Frankowski and Vert, 2020; Ma *et al.*, 2020; Matsumoto *et al.*, 2017). RVAgene could supplement such methods either by providing denoised input data, or by completely replacing the temporal ordering/differential equation-based components of these methods (which can be notoriously difficult to parameterize) with data produced from an RVAgene generative model of the gene expression dynamics.

RVAgene is currently agnostic of irregular time intervals between consecutive points in a time series, i.e. it standardizes the time interval. This is not usually a concern for single-cell data, since with pseudotime information we can choose appropriate time intervals. However, in other cases, such as in response to kidney injury (Liu *et al.*, 2017), standardizing time intervals distorts the dynamic profiles. Since RVAgene seeks to describe broad temporal patterns, we do not see this as a critical issue, though it would be desirable to generalize the model. A simple way to model irregularly spaced time points would be to augment the data through interpolation, though this is difficult without making strong assumptions about the (generally unknown) noise model. Gaussian process models (Hensman

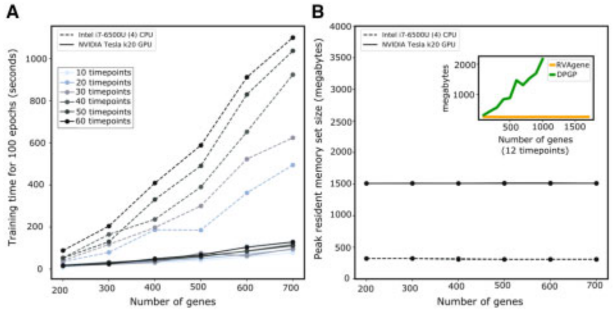

Fig. 6. Training RVAgene is reasonably scalable on CPU and even more so using hardware acceleration through GPU. (**A**) Time cost of training RVAgene for 100 epochs for datasets with varying number of genes and time points on CPU and GPU; data from Klein *et al.* (2015). (**B**) Maximum memory utilized during training of the model on CPU an GPU for the cases in (A), inset plot: comparison of max memory used compared to DPGP for varying number of genes

et al., 2013; McDowell et al., 2018) can take irregular data as input, although (as noted above) are not efficient enough to run on large datasets. An alternative approach would be to modify the recurrent network architecture to take time points explicitly as input values, this would enable modeling of irregular or asynchronous data (Wu et al., 2018).

RVAgene models in discrete time steps. There is no simple modification to the recurrent network structure that allows for prediction on continuously valued time. However, a recent development: neural ordinary differential equations (ODEs) (Chen et al., 2018), enables modeling of time series data with continuous timepoints. Chen et al. (2018) describe a generative latent ODE architecture similar to that of RVAgene, except that in their case the recurrent decoder network is replaced by a neural ODE decoder network. Chen et al. (2018) demonstrate accurate results using synthetic data, however when we applied the method to the ESC single-cell differentiation dataset (Klein et al., 2015), the neural ODE network was found to converge very slowly and was overall underfit (Supplementary Fig. S9). The latent ODE method used by Chen et al. (2018) does not address the challenge of modeling asynchronous/irregularly spaced data, but this has been more recently addressed (Rubanova et al., 2019). These new models may well lead to future improvements in network architectures, although it seems that computational progress is needed before they can be successfully applied to complex biological systems.

In the current work, the prior on latent space used throughout was a unit spherical Normal, appropriate for exploratory data analysis where we have no further knowledge about structure in the latent space. However, given more information, e.g. that the data contains *k* clusters, a different prior on the latent space might be more appropriate. A multimodal prior—such as a Gaussian Mixture Model (GMM) prior—would permit structured (multi-modal) representations. However, the KL-divergence for an arbitrary GMM is not tractable; approximation (Hershey and Olsen, 2007) or numerical computation would be necessary. Moreover, there is a greater problem: mixture models contain discrete parameters and VAE models are ill-suited for the optimization of discrete parameters (Dilokthanakul et al., 2016), thus directly replacing the Normal prior of a VAE with a GMM is not feasible. A workaround to this problem is presented in (Dilokthanakul et al., 2016), however implementing this for a recurrent model architecture remains an open problem.

The points raised above offer much scope for future work. These include the design of new latent space models with informative priors, modeling irregular time series data and modeling in continuous time. Developments in some of these areas (Chen et al., 2018), while promising, tend to rely on training data with relatively low levels of noise: far from the reality of most biological data. Thus it seems highly likely to be beneficial for both machine learning and biology to develop new neural network architectures in light of biological data.

Acknowledgements

The authors were grateful to A.P. McMahon for valuable discussions and comments on the manuscript.

Funding

This work was partially supported by an Andrew J. Viterbi Fellowship in Computational Biology and Bioinformatics (to R.M.) and by the National Science Foundation (DMS 2045327 to A.L.M.).

Conflict of Interest: none declared.

References

Ainsworth,S.K. *et al.* (2018) oi-VAE: output interpretable VAEs for nonlinear group factor analysis. In: *International Conference on Machine Learning*, Stockholmsmässan, Stockholm Sweden, pp. 119–128.

Aubin-Frankowski,P.-C. and Vert,J.-P. (2020) Gene regulation inference from single-cell RNA-seq data with linear differential equations and velocity inference. *Bioinformatics*, **36**, 4774–4780.

Barron,A.R. (1994) Approximation and estimation bounds for artificial neural networks. *Mach. Learn.*, **14**, 115–133.

Botchkarev,A. (2018) Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology. *arXiv preprint arXiv:1809.03006*.

Bouchard-Côté,A. and Jordan,M.I. (2010) Variational inference over combinatorial spaces. In: *Advances in Neural Information Processing Systems*, pp. 280–288.

Chen,S. and Mar,J.C. (2018) Evaluating methods of inferring gene regulatory networks highlights their lack of performance for single cell gene expression data. *BMC Bioinformatics*, **19**, 232.

Chen,T.Q. *et al.* (2018) Neural ordinary differential equations. In: *Advances in Neural Information Processing Systems*, Montreal Convention Centre, Montreal, Canada, pp. 6571–6583.

Ching,T. *et al.* (2018) Opportunities and obstacles for deep learning in biology and medicine. *J. R. Soc. Interface*, **15**, 20170387.

Cooke,E.J. *et al.* (2011) Bayesian hierarchical clustering for microarray time series data with replicates and outlier measurements. *BMC Bioinformatics*, **12**, 399.

Cybenko,G. (1989) Approximation by superpositions of a sigmoidal function. *Math. Control Sign. Syst.*, **2**, 303–314.

Deng,Y. *et al.* (2019) Scalable analysis of cell-type composition from single-cell transcriptomics using deep recurrent learning. *Nat. Methods*, **16**, 311–314.

Deshpande,A. *et al.* (2019) Network inference with granger causality ensembles on single-cell transcriptomic data. *bioRxiv preprint bioRxiv:534834*.

Dilokthanakul,N. *et al.* (2016) Deep unsupervised clustering with Gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*.

Ding,J. *et al.* (2018) Interpretable dimensionality reduction of single cell transcriptome data with deep generative models. *Nat. Commun.*, **9**, 1–13.

Eraslan,G. *et al.* (2019) Single-cell RNA-seq denoising using a deep count autoencoder. *Nat. Commun.*, **10**, 1–14.

Fabius,O. and van Amersfoort,J.R. (2014) Variational recurrent auto-encoders. *In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, Workshop Track Proceedings*.

Ferguson,T.S. (1973) A Bayesian analysis of some nonparametric problems. *Ann. Stat.*, **1**, 209–230.

Funahashi,K.-I. (1989) On the approximate realization of continuous mappings by neural networks. *Neural networks*, **2**, 183–192.

Haghverdi,L. *et al.* (2016) Diffusion pseudotime robustly reconstructs lineage branching. *Nat. Methods*, **13**, 845–848.

Harris,C.R. *et al.* (2020) Array programming with numpy. *Nature*, **585**, 357–362.

Hensman,J. *et al.* (2013) Hierarchical Bayesian modelling of gene expression time series across irregularly sampled replicates and clusters. *BMC Bioinformatics*, **14**, 252.

Hershey,J.R. and Olsen,P.A. (2007) Approximating the Kullback Leibler divergence between Gaussian mixture models. In: *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, Hawai'i Convention Center, Honolulu, Vol. 4. IEEE, pp. IV–317.

Higgins,I. *et al.* (2016) beta-VAE: Learning basic visual concepts with a constrained variational framework. In: *International Conference on Learning Representations, ICLR 2017, San Juan, Puerto Rico, Poster Track Proceedings*.

Hinton,G.E. and Salakhutdinov,R.R. (2006) Reducing the dimensionality of data with neural networks. *Science*, **313**, 504–507.

Hochreiter,S. and Schmidhuber,J. (1997) Long short-term memory. *Neural Comput.*, **9**, 1735–1780.

Hoffman,M.D. *et al.* (2013) Stochastic variational inference. *J. Mach. Learn. Res.*, **14**, 1303–1347.

Hornik,K. *et al.* (1989) Multilayer feedforward networks are universal approximators. *Neural Netw.*, **2**, 359–366.

Ingraham,J. and Marks,D. (2017) Variational inference for sparse and undirected models. In: *International Conference on Machine Learning*. PMLR, Stockholmsmässan, Stockholm Sweden, pp. 1607–1616.

Jang,S. *et al.* (2017) Dynamics of embryonic stem cell differentiation inferred from single-cell transcriptomics show a series of transitions through discrete cell states. *eLife*, **6**, e20487.

Jetka,T. *et al.* (2018) An information-theoretic framework for deciphering pleiotropic and noisy biochemical signaling. *Nat. Commun.*, **9**, 1–9.

Kim,J. *et al.* (2021) TENET: gene network reconstruction using transfer entropy reveals key regulatory factors from single cell transcriptomic data. *Nucleic Acids Res.*, **49**, e1.

Kingma,D.P. and Welling,M. (2014) Auto-encoding variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, Canada - Conference Track Proceedings*.

Klein,A.M. *et al.* (2015) Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell*, **161**, 1187–1201.

Lin,C. *et al.* (2017) Using neural networks for reducing the dimensions of single-cell RNA-seq data. *Nucleic Acids Res.*, **45**, e156.

Liu,J. *et al.* (2017) Molecular characterization of the transition from acute to chronic kidney injury following ischemia/reperfusion. *JCI Insight*, **2**, e9471.

Lopez,R. *et al.* (2018) Deep generative modeling for single-cell transcriptomics. *Nat. Methods*, **15**, 1053–1058.

Ma,B. *et al.* (2020) Inference of gene regulatory networks based on nonlinear ordinary differential equations. *Bioinformatics*, **36**, 4885–4893.

Malhotra,P. *et al.* (2015) Long short term memory networks for anomaly detection in time series. In *23rd European Symposium on Artificial Neural Networks, ESANN 2015, Bruges, Belgium*, *Proceedings*, Vol. 89. Presses Universitaires de Louvain, pp. 89–94.

Matsumoto,H. *et al.* (2017) SCODE: an efficient regulatory network inference algorithm from single-cell RNA-Seq during differentiation. *Bioinformatics*, **33**, 2314–2321.

McDowell,I.C. *et al.* (2018) Clustering gene expression time series data using an infinite Gaussian process mixture model. *PLoS Comput. Biol.*, **14**, e1005896.

Nallapati,R. *et al.* (2016) Abstractive text summarization using sequence-to-sequence RNNs and beyond. *In: The SIGNLL Conference on Computational Natural Language Learning (CoNLL), Berlin, Germany, 2016.*

Neugarten,J. *et al.* (2000) Effect of gender on the progression of nondiabetic renal disease: a meta-analysis. *J. Am. Soc. Nephrol.*, **11**, 319–329.

Raj,A. and Van Oudenaarden,A. (2008) Nature, nurture, or chance: stochastic gene expression and its consequences. *Cell*, **135**, 216–226.

Ransick,A. *et al.* (2019) Single-cell profiling reveals sex, lineage, and regional diversity in the mouse kidney. *Dev. Cell*, **51**, 399–413.e7.

Rubanova,Y. *et al.* (2019) Latent ordinary differential equations for irregularly-sampled time series. In: *Advances in Neural Information Processing Systems*, Vancouver, BC, Canada, pp. 5321–5331.

Rumelhart,D.E. *et al.* (1986) Learning representations by back-propagating errors. *Nature*, **323**, 533–536.

Saelens,W. *et al.* (2019) A comparison of single-cell trajectory inference methods. *Nat. Biotechnol.*, **37**, 547–554.

Svensson,V. *et al.* (2018) Exponential scaling of single-cell RNA-seq in the past decade. *Nat. Protoc.*, **13**, 599–604.

Svensson,V. *et al.* (2020) Interpretable factor models of single-cell RNA-seq via variational autoencoders. *Bioinformatics*, **36**, 3418–3421.

Talwar,D. *et al.* (2018) Autoimpute: autoencoder based imputation of single-cell RNA-seq data. *Sci. Rep.*, **8**, 1–11.

Wang,D. and Gu,J. (2018) VASC: dimension reduction and visualization of single-cell RNA-seq data by deep variational autoencoder. *Genomics Proteomics Bioinf.*, **16**, 320–331.

Wang,J. *et al.* (2019) Data denoising with transfer learning in single-cell transcriptomics. *Nat. Methods*, **16**, 875–878.

Way,G.P. and Greene,C.S. (2017) Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders. In: *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, Fairmont Orchid, Hawaii, Vol. 23 2018: pp. 80-91.

Wu,S. *et al.* (2018) Modeling asynchronous event sequences with RNNs. *J. Biomed. Inf.*, **83**, 167–177.

Zhang,C. *et al.* (2019) Advances in variational inference. *IEEE Trans. Pattern Anal. Mach. Intell.*, **41**, 2008–2026.

Zhu,L. *et al.* (2019) Semisoft clustering of single-cell data. *Proc. Natl. Acad. Sci. USA*, **116**, 466–471.