A Appendix

A.1 Sampling from L^p-nested Symmetric Distributions

We reproduce the sampling algorithm for L^p-nested symmetric distributions from Sinz and Bethge (2010) in Alg. 1.

Input : The radial distribution $\psi_0(v_0)$ of an L^p-nested symmetric distribution p_{L^p} for the L^p-nested function f

Output : Sample x from p_{L^p}

1. Sample v_0 from a beta distribution $\beta[n, 1]$
2. For each inner node i of the tree associated with f, sample the auxiliary variable s_i from a Dirichlet distribution Dir $\left[\frac{n_{i,1}}{p_i}, \ldots, \frac{n_{i,l_1}}{p_i}\right]$ where $n_{i,k}$ are the number of leaves in the subtree under node i, k. Obtain coordinates on the L^p-nested sphere within the positive orthant by $\boldsymbol{s}_i \mapsto \boldsymbol{s}_i^{\frac{1}{p_i}} = \tilde{\boldsymbol{u}}_i$ (the exponentiation is taken component-wise)
3. Transform these samples to Cartesian coordinates by $v_i \cdot \tilde{\boldsymbol{u}}_i = \boldsymbol{v}_{i,1:l_i}$ for each inner node, starting from the root node and descending to lower layers. The components of $\boldsymbol{v}_{i,1:l_i}$ constitute the radii for the layer direct below them. If $i = 0$, the radius had been sampled in step 1
4. Once the two previous steps have been repeated until no inner node is left, we have a sample $\boldsymbol{x}$ from the uniform distribution in the positive quadrant. Normalize $\boldsymbol{x}$ to get a uniform sample from the sphere $\boldsymbol{u} = \frac{\boldsymbol{x}}{f(\boldsymbol{x})}$
5. Sample a new radius $\tilde{v}_0$ from the radial distribution of the target radial distribution ψ_0 and obtain the sample via $\tilde{\boldsymbol{x}} = \tilde{v}_0 \cdot \boldsymbol{u}$
6. Multiply each entry x_i of $\tilde{\boldsymbol{x}}$ by and independent sample z_i from the uniform distribution over $\{-1, 1\}$.

Algorithm 1: Exact sampling algorithm for L^p-nested symmetric distributions from Sinz and Bethge (2010)

A.2 Learned Exponents

An interesting question when learning the exponents of the prior is, if the trivial case is learned, in which all the exponents become equal to p_0. This implies a fully factorized prior over all latent variables. To explore this we set $p_0 = 2.1$ and initialize the exponents of the subspaces to $p_{1,\ldots,3} = 2.0$. We train 15 models for each value of $\beta \in \{0.5, 1.0, 2.0, 3.0, 4.0, 5.0\}$. Fig. 7 depicts histograms of the learnt exponents, where we sort the exponents such that $p_1 < p_2 < p_3$. Interestingly, the exponents with highest frequency are 1.95 for p_1, 1.98 for p_2, and 2.17 for p_3. Also, all values of p_3 are strictly larger than 2.17, meaning that these exponents are also always different from p_0. This small deviation from the Gaussian with $p = 2.0$ seems to be sufficient to break symmetry and produce a more structured representation.

Table 1: Hyperparameters of FactorVAE, β-VAE, β-TCVAE, and ISA-VAE evaluated on the dSprites dataset. We evaluated each model on the whole range of regularization strength parameters. (γ for FactorVAE and β for the other models)

Parameter	**Values**
β	[1, 1.5, 2, 2.5, 3, 3.5, 4, 5, 6]
γ (FactorVAE)	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Epochs	20
Learning Rate	0.001
Batch Size	2048
Latent dimension	20
Loss function	Bernoulli
Optimizer	Adam

Table 2: Hyperparameters of β-VAE, β-TCVAE, and ISA-VAE evaluated on the 3D faces dataset (Paysan et al., 2009). We evaluated each model on the whole range of regularization strength parameters.

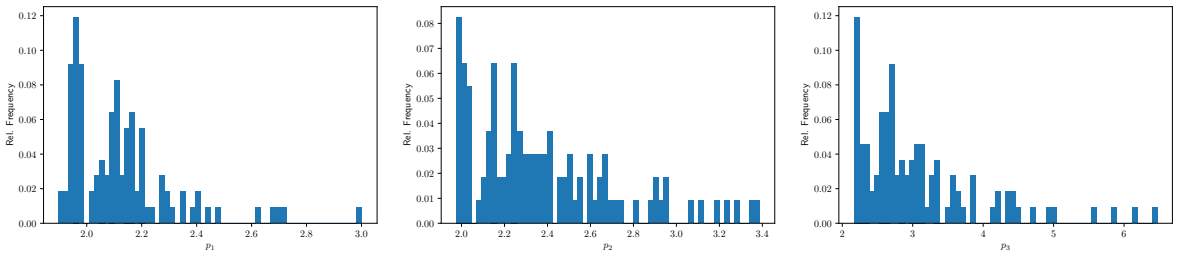
Parameter	**Values**
β	[1, 1.5, 2, 2.5, 3, 4]
Epochs	1500
Learning Rate	0.001
Batch Size	2048
Latent dimension	10
Loss function	Bernoulli
Optimizer	Adam

A.3 Hyperparameters

The hyperparameters that we use for the experiments on the dSprites dataset can be found in table 1, and the hyperparameters for the experiments on the 3D faces dataset in table 2.

A.4 Model Architecture (PyTorch)

The models were trained with the optimization algorithm Adam (Kingma and Ba, 2015) using a learning rate parameter of 0.001



(a) Histogram of p_1. (b) Histogram of p_2. (c) Histogram of p_3.

Figure 7: Histogram of learned exponents on the 3d faces dataset. To identify the different subspaces we choose the ordering $p_1 < p_2 < p_3$.

All unmentioned hyperparameters are PyTorch v0.41 defaults.

```
class MLPEncoder(nn.Module):
  def __init__(self, output_dim):
    super(MLPEncoder, self).__init__()
    self.output_dim = output_dim

    self.fc1 = nn.Linear(4096,1200)
    self.fc2 = nn.Linear(1200,1200)
    self.fc3 = nn.Linear(1200,output_dim)

    self.conv_z=nn.Conv2d(64,output_dim,4,1,0)

    self.act = nn.ReLU(inplace=True)

  def forward(self, x):
    h = x.view(-1, 64 * 64)
    h = self.act(self.fc1(h))
    h = self.act(self.fc2(h))
    h = self.fc3(h)
    z = h.view(x.size(0), self.output_dim)
    return z


class MLPDecoder(nn.Module):
  def __init__(self, input_dim):
    super(MLPDecoder, self).__init__()
    self.net = nn.Sequential(
      nn.Linear(input_dim, 1200),
      nn.Tanh(),
      nn.Linear(1200, 1200),
      nn.Tanh(),
      nn.Linear(1200, 1200),
      nn.Tanh(),
      nn.Linear(1200, 4096)
    )

  def forward(self, z):
    h = z.view(z.size(0), -1)
    h = self.net(h)
    mu_img = h.view(z.size(0), 1, 64, 64)
    return mu_img
```

Architecture of the encoder and decoder which is identical to the architecture in Chen et al. (2018).

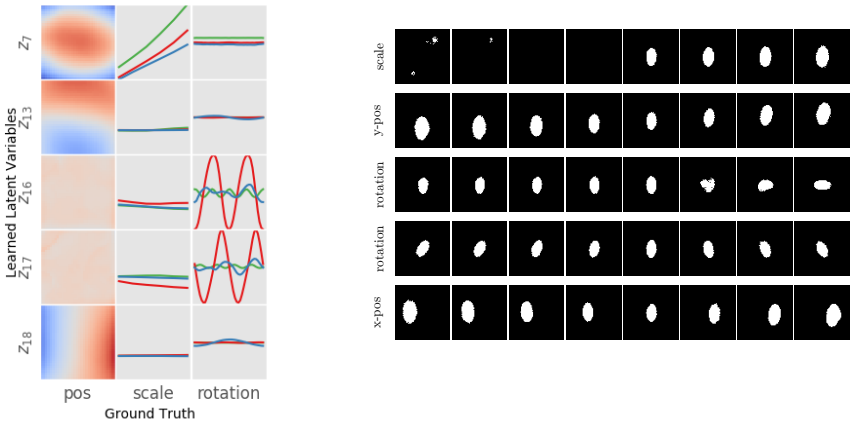
```
class Discriminator(nn.Module):
    def __init__(self, z_dim):
        super(Discriminator, self).__init__()
        self.net = nn.Sequential(
            nn.Linear(z_dim, 1000),
            nn.LeakyReLU(0.2, True),
            nn.Linear(1000, 1000),
            nn.LeakyReLU(0.2, True),
            nn.Linear(1000, 1000),
            nn.LeakyReLU(0.2, True),
            nn.Linear(1000, 1000),
            nn.LeakyReLU(0.2, True),
            nn.Linear(1000, 1000),
            nn.LeakyReLU(0.2, True),
            nn.Linear(1000, 2),
        )

    def forward(self, z):
        return self.net(z).squeeze()
```

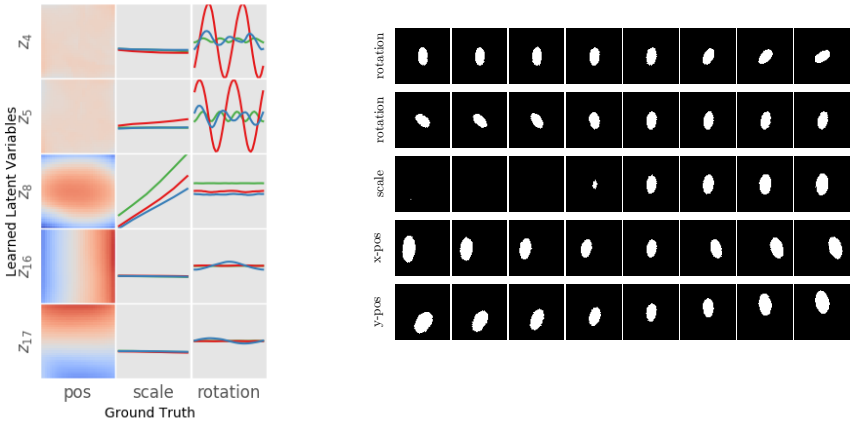
Architecture of the discriminator of FactorVAE.

A.5 Disentangled Representations and Latent Traversals

We use the plotting technique established in Chen et al. (2018) for visualizing latent representations and additionally show images generated by traversals of the latent along the respective axis. The red and blue colour coding in the first column denotes the value of the latent variable for the respective x,y-coordinate of the sprite in the image. Coloured lines indicate the object shape with red for ellipse, green for square, and blue for heart. We observed that the MIG scores after training are usually bimodal: Either a model disentangles well or it does not reach a good MIG score. Therefore, to choose a representative model for each model class we take the average of the upper 50% quantile of MIG scores and choose a representative model that minimizes the mahalanobis distance, defined by mean and variance of MIG score and reconstruction loss. ISA-layout: ISA-VAE: $l_0 = 5$, $l_{1,\dots,5} = 5$, $p_0 = 2.1$, $p_{1,\dots,5} = 2.2$.

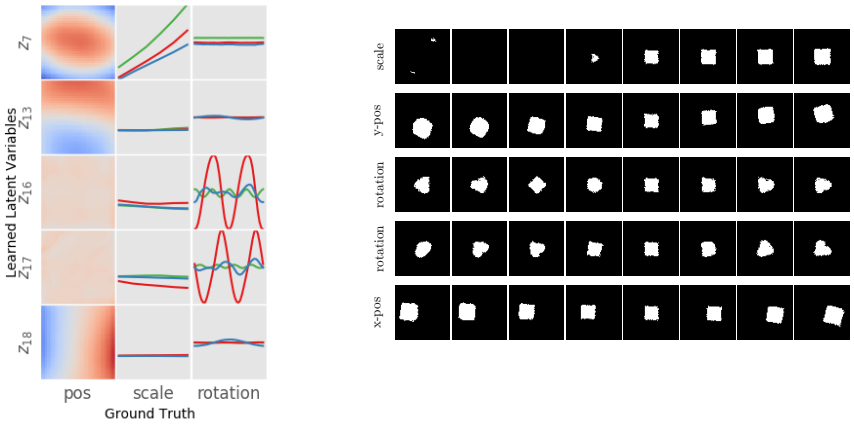


(a) β-VAE, $\beta = 1.0$, MIG: $0.14, logpx : -21.99$

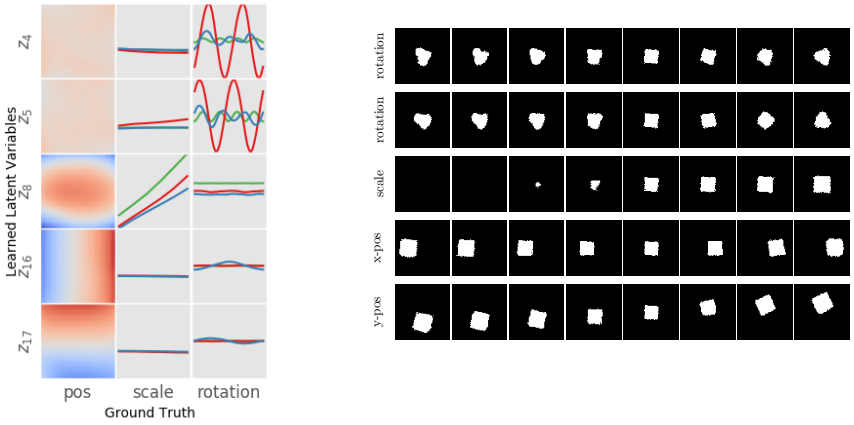


(b) ISA-VAE, $\beta = 1.0$, MIG: 0.20, logpx: -20.93

Figure 8: Disentangled representations for representative models of the upper quantile of MIG scores for β-VAE (identical with β-TCVAE for $\beta = 0$) and ISA-VAE (ISA-TCVAE identical for $\beta = 0$) and latent traversals for the ellipse shape.

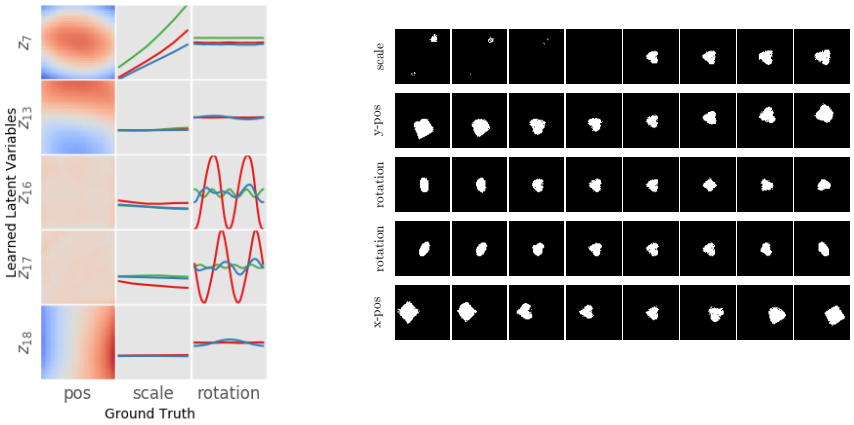


(a) β-VAE, $\beta = 1.0$, MIG: $0.14, logpx : -21.99$

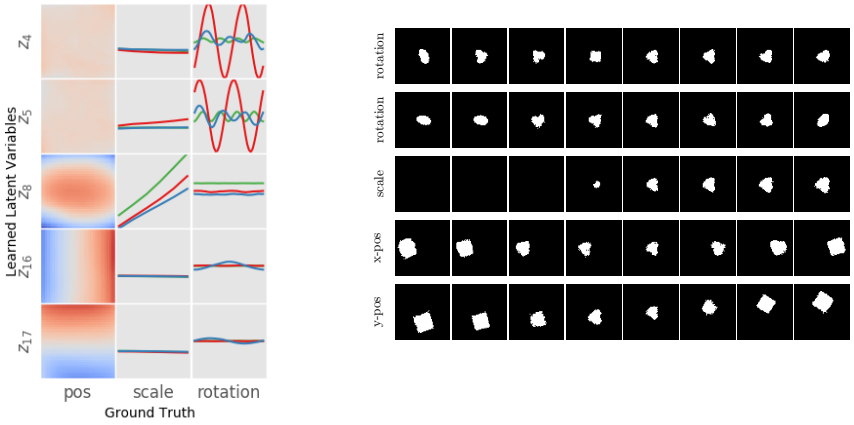


(b) ISA-VAE, $\beta = 1.0$, MIG: 0.20, logpx: -20.93

Figure 9: Disentangled representations for representative models of the upper quantile of MIG scores for β-VAE (identical with β-TCVAE for $\beta = 0$) and ISA-VAE (ISA-TCVAE identical for $\beta = 0$) and latent traversals for the square shape.

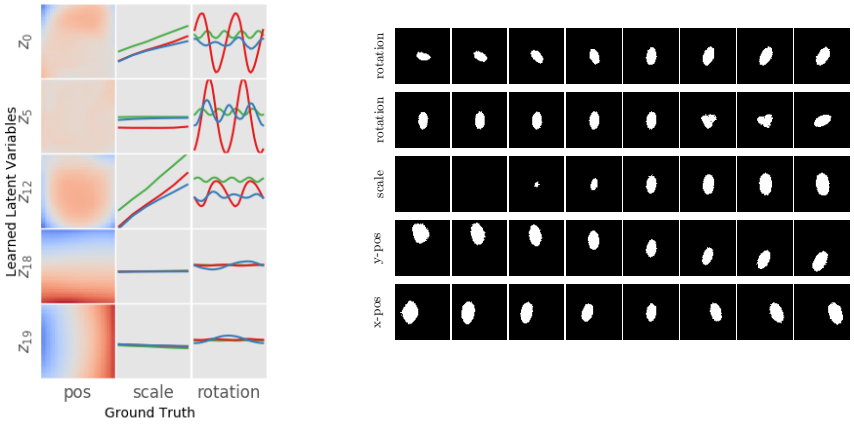


(a) β-VAE, $\beta = 1.0$, MIG: $0.14, logpx : -21.99$

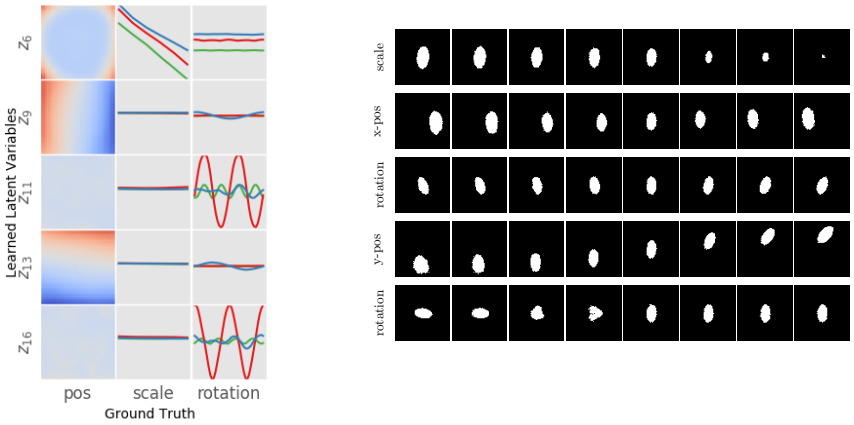


(b) ISA-VAE, $\beta = 1.0$, MIG: 0.20, logpx: -20.93

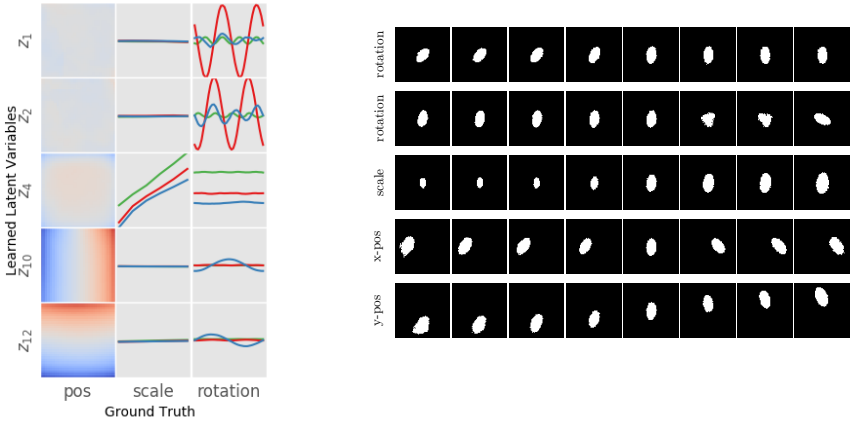
Figure 10: Disentangled representations for representative models of the upper quantile of MIG scores for β-VAE (identical with β-TCVAE for $\beta = 0$) and ISA-VAE (ISA-TCVAE identical for $\beta = 0$) and latent traversals for the heart shape.



(a) β-VAE, $\beta = 2.0$, MIG: 0.28, logpx: -29.40

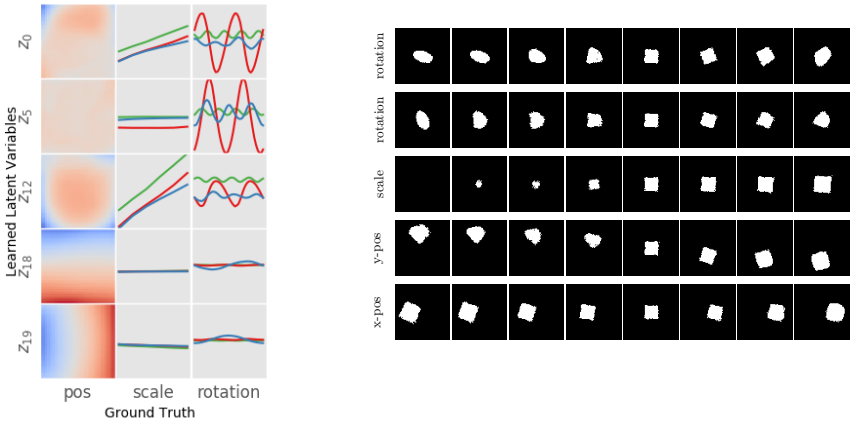


(b) β-TCVAE, $\beta = 2.0$, MIG: 0.30, logpx: -27.15

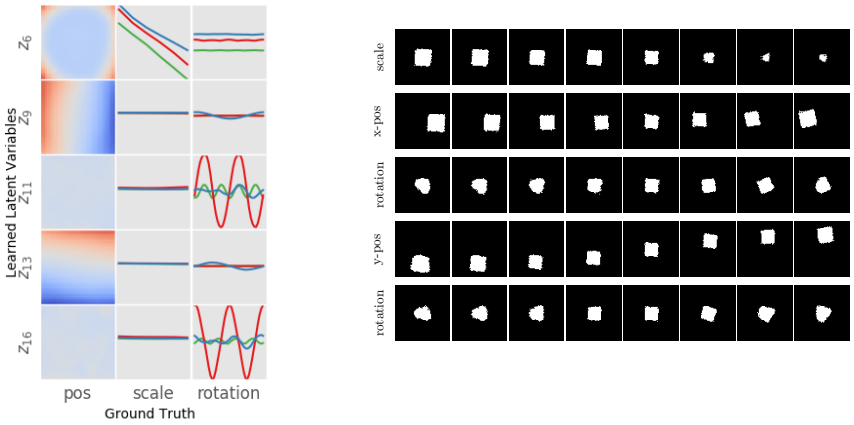


(c) ISA-VAE, $\beta = 2.0$, MIG: 0.41, logpx: -25.86

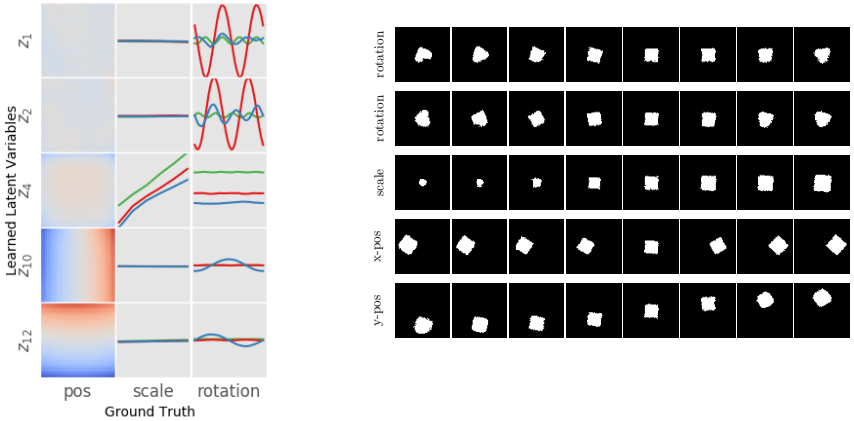
Figure 11: Disentangled representations for models representative for the upper quantile of MIG scores for $\beta = 2.0$ for β-VAE, β-TCVAE, ISA-VAE and ISA-TCVAE and latent traversals for the ellipse shape.



(a) β-VAE, $\beta = 2.0$, MIG: 0.28, logpx: -29.40

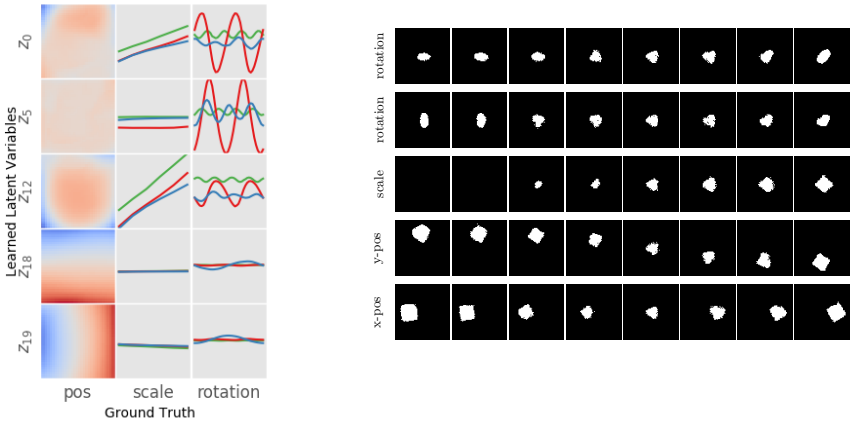


(b) β-TCVAE, $\beta = 2.0$, MIG: 0.30, logpx: -27.15

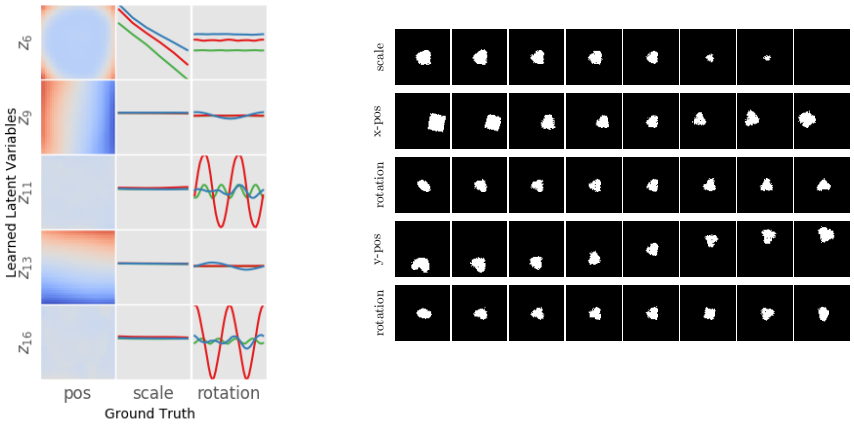


(c) ISA-VAE, $\beta = 2.0$, MIG: 0.41, logpx: -25.86

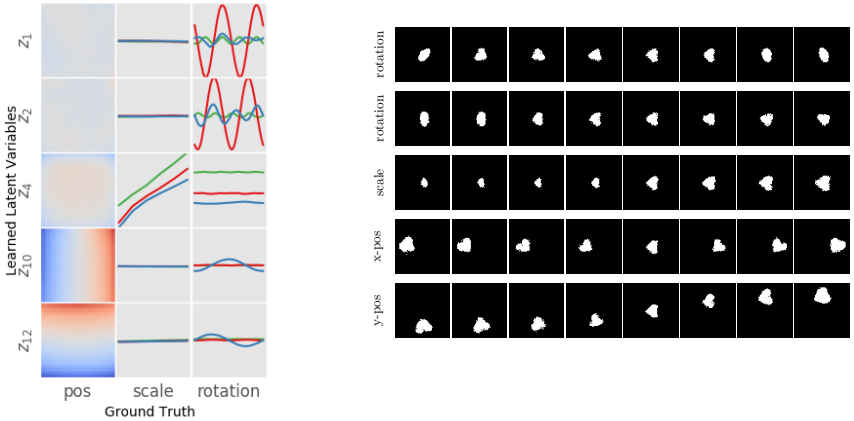
Figure 12: Disentangled representations for models representative for the upper quantile of MIG scores for $\beta = 2.0$ for β-VAE, β-TCVAE, ISA-VAE and ISA-TCVAE and latent traversals for the square shape.



(a) β-VAE, $\beta = 2.0$, MIG: 0.28, logpx: -29.40

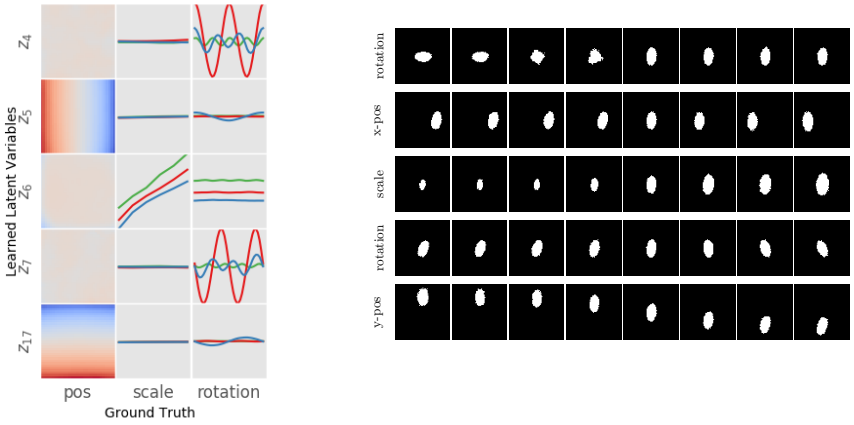


(b) β-TCVAE, $\beta = 2.0$, MIG: 0.30, logpx: -27.15

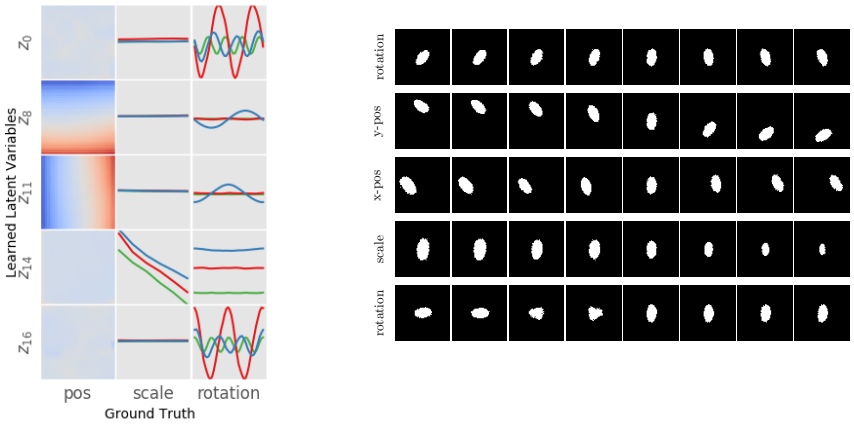


(c) ISA-VAE, $\beta = 2.0$, MIG: 0.41, logpx: -25.86

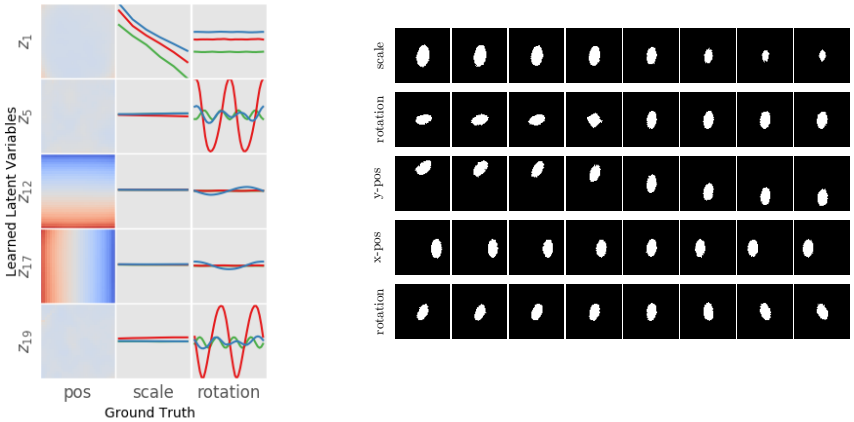
Figure 13: Disentangled representations for models representative for the upper quantile of MIG scores for $\beta = 2.0$ for β-VAE, β-TCVAE, ISA-VAE and ISA-TCVAE and latent traversals for the heart shape.



(a) β-VAE, $\beta = 3.0$, MIG: 0.47, $logpx$: -33.44

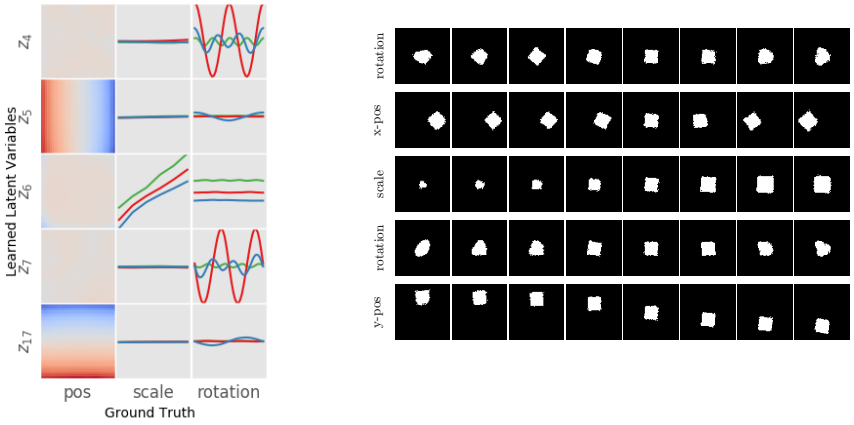


(b) β-TCVAE, $\beta = 3.0$, MIG: 0.43, logpx: -33.40

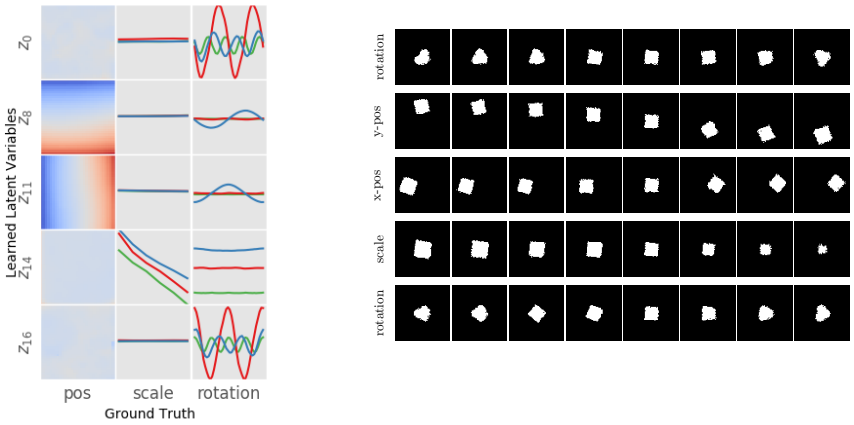


(c) ISA-VAE, $\beta = 3.0$, MIG: 0.48, logpx: -32.42

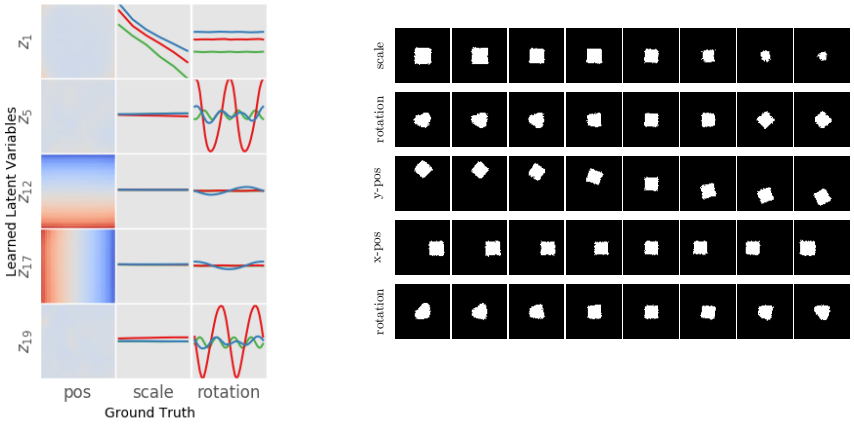
Figure 14: Disentangled representations for models representative for the upper quantile of MIG scores for $\beta = 3.0$ for β-VAE, β-TCVAE, ISA-VAE and ISA-TCVAE and latent traversals for the ellipse shape.



(a) β-VAE, $\beta = 3.0$, MIG: 0.47, $logpx$: -33.44

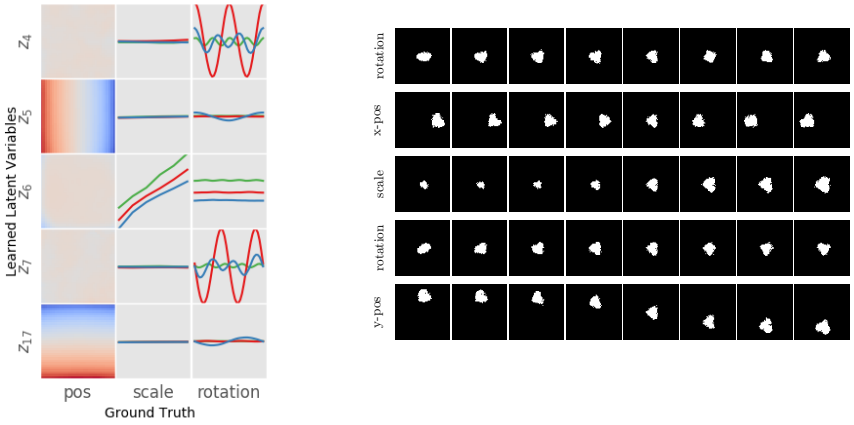


(b) β-TCVAE, $\beta = 3.0$, MIG: 0.43, logpx: -33.40

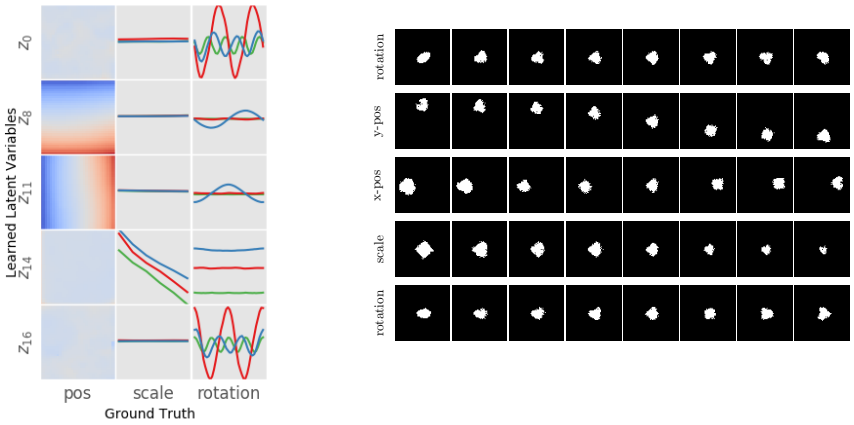


(c) ISA-VAE, $\beta = 3.0$, MIG: 0.48, logpx: -32.42

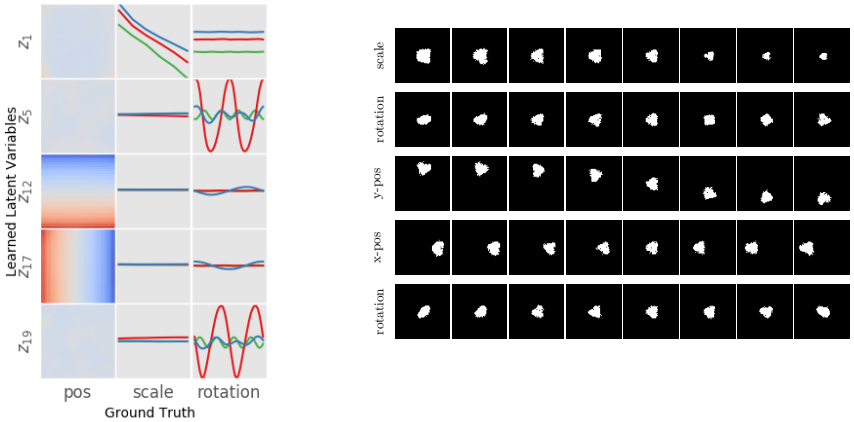
Figure 15: Disentangled representations for models representative for the upper quantile of MIG scores for $\beta = 3.0$ for β-VAE, β-TCVAE, ISA-VAE and ISA-TCVAE and latent traversals for the square shape.



(a) β-VAE, $\beta = 3.0$, MIG: 0.47, $logpx$: -33.44



(b) β-TCVAE, $\beta = 3.0$, MIG: 0.43, logpx: -33.40



(c) ISA-VAE, $\beta = 3.0$, MIG: 0.48, logpx: -32.42

Figure 16: Disentangled representations for models representative for the upper quantile of MIG scores for $\beta = 3.0$ for β-VAE, β-TCVAE, ISA-VAE and ISA-TCVAE and latent traversals for the heart shape.

B Toy examples showing biases in Variational Inference and β-Variational Inference

This section provides the details of the toy examples that reveal the biases in variational methods.

First we will consider the factor analysis model showing that Variational Inference (VI) breaks the degeneracy of the maximum-likelihood solution to 1) discover orthogonal weights that lie in the PCA directions, 2) prune out extra components in over-complete factor analysis models, even though there are solutions with the same likelihood that preserve all components. We also show that in these examples the β-VI returns identical model fits to VI regardless of the setting of β.

Second, we consider an over-complete ICA model and initialize using the true model. We show that 1) VI is biased away from the true component directions towards more orthogonal directions, and 2) β-VI with a modest setting of $\beta = 5$ prunes away one of the components and finds orthogonal directions for the other two. That is, it finds a disentangled representation, but one which does not reflect the underlying components.

B.1 Background

The β-VAE optimizes the modified free-energy, $\mathcal{F}_\beta(q(z_{1:N}), \theta)$, with respect to the parameters θ and the variational approximation $q(z_{1:N})$,

$$\begin{aligned}\mathcal{F}_\beta(q(z_{1:N}),\theta) =& \mathbb{E}_{q(z_{1:N})}(\log p(x_{1:N}|z_{1:N},\theta)) \\ & - \beta \mathrm{KL}(q(z_{1:N})||p(z_{1:N})).\end{aligned} \tag{13}$$

Consider the case where $M = \frac{1}{\beta}$ is a positive integer, $M \in \mathbb{N}$, we then have

$$\begin{aligned}\mathcal{F}_\beta(q(z_{1:N}),\theta) = \sum_{n=1}^{N} \Big[& \mathbb{E}_{q(z_n)}(M(\beta)\log p(x_n|z_n,\theta)) \\ & - \mathrm{KL}(q(z_n)||p(z_n))\Big]\end{aligned}$$

In this case, the β-VAE can be thought of as attaching M replicated observations to each latent variable z_n and then running standard variational inference on the new replicated dataset. This can equivalently be thought of as raising each likelihood $p(x_n|z_n,\theta)$ to the power M.

Now in real applications β will be set to a value that is greater than one. In this case, the effect of β is the opposite: it is to reduce the number of effective data points per latent variable to be less than one $M < 1$. Or equivalently we raise each likelihood term to a power M that is less than one. Standard VI is then run on these modified data (e.g. via joint optimization of q and θ).

Although this view is mathematically straightforward, the perspective of the β-VAE i) modifying the dataset, and ii) applying standard VI, is useful as it will allow us to derive optimal solutions for the variational distribution $q(z)$ in simple cases like the factor analysis model considered next.

B.2 Factor analysis

Consider the factor analysis generative model. Let $\mathbf{x} \in \mathbb{R}^L$ and $\mathbf{z} \in \mathbb{R}^K$.

$$\begin{aligned}&\text{for } \; n = 1...N \\ &\quad \mathbf{z}_n \sim \mathcal{N}(\mathbf{0}, \mathrm{I}), \\ &\quad \mathbf{x}_n \sim \mathcal{N}(W\mathbf{z}_n, D) \;\; \text{where} \;\; D = \mathrm{diag}([\sigma_1^2, ..., \sigma_{\mathrm{D}}^2])\end{aligned} \tag{14}$$

The true posterior is a Gaussian $p(\mathbf{z}_n|\mathbf{x}_n,\theta) = \mathcal{N}(\mathbf{z}; \mu_{\mathbf{z}|\mathbf{x}}, \Sigma_{\mathbf{z}|\mathbf{x}})$ where

$$\begin{aligned}\mu_{\mathbf{z}|\mathbf{x}} &= \Sigma_{\mathbf{z}|\mathbf{x}} W^\top D^{-1}\mathbf{x} \\ \text{and} \;\; \Sigma_{\mathbf{z}|\mathbf{x}} &= (W^\top D^{-1} W + \mathrm{I})^{-1}.\end{aligned} \tag{15}$$

The true log-likelihood of the parameters is

$$\begin{aligned}\log p(\mathbf{x}_{1:N}|\theta) = & \sum_{n=1}^{N} \log \mathcal{N}(\mathbf{x}_n, \mathbf{0}, WW^\top + D) \\ = & -\frac{N}{2}\log\det(2\pi(WW^\top + D)) \\ & -\frac{1}{2}\sum_{n=1}^{N} \mathbf{x}_n^\top (WW^\top + D)^{-1}\mathbf{x}_n \\ = & -\frac{1}{2}N\Big[\log\det(2\pi(WW^\top + D)) \\ & + \mathrm{trace}((WW^\top + D)^{-1}(\mu_{\mathbf{x}}\mu_{\mathbf{x}}^\top + \Sigma_{\mathbf{x}}))\Big]\end{aligned}$$

Here we have defined the empirical mean and covariance of the observations $\mu_{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$ and $\Sigma_{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \mu_{\mathbf{x}})(\mathbf{x}_n - \mu_{\mathbf{x}})^\top$ i.e. the sufficient statistics.

The likelihood is invariant under orthogonal transformations of the latent variables: $\mathbf{z}' = R\mathbf{z}$ where $RR^\top = \mathrm{I}$.

Interpreting β-VI as running VI in a modified generative model (see previous section) we have the new generative process

$$\begin{aligned}&\text{for } \; n = 1...N \\ &\quad \mathbf{z}_n \sim \mathcal{N}(\mathbf{z}_n; \mathbf{0}, \mathrm{I}), \\ &\quad \text{for } \; m = 1...M(\beta) \\ &\qquad \mathbf{x}_{n,m} \sim \mathcal{N}(W\mathbf{z}_n, D) \;\; \text{where} \;\; D = \mathrm{diag}([\sigma_1^2, ..., \sigma_{\mathrm{D}}^2])\end{aligned}$$

We now observe data and set $\mathbf{x}_{n,m} = \mathbf{x}_n$.

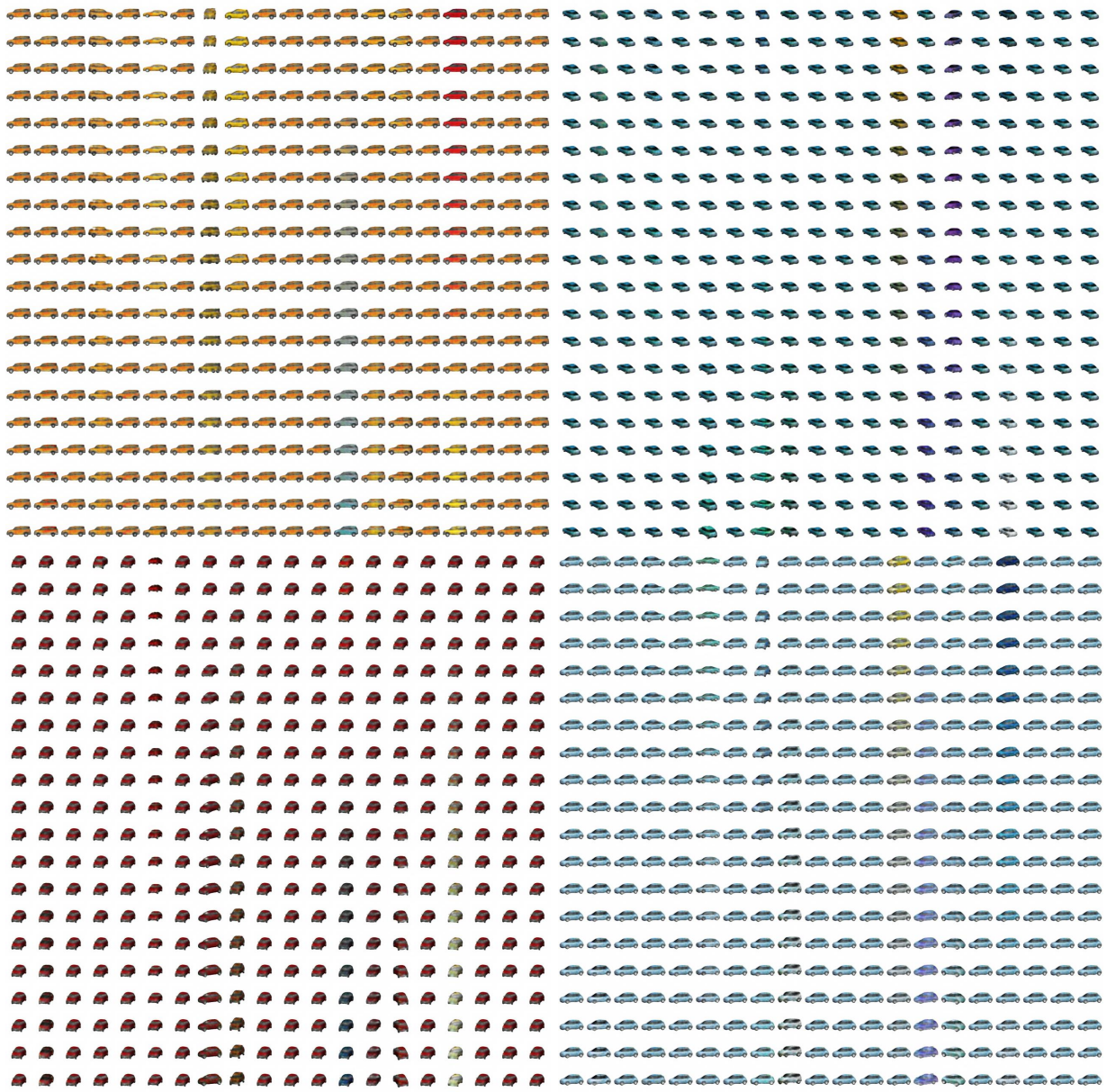


Figure 17: Latent traversals on the cars3d dataset (Reed et al., 2015), columns correspond to individual latent dimensions, rows are traversals along the respective latent component. Across different shapes of cars, the latent variables consistently encode features such as azimuth (8th column), elevation (15th column), and color (e.g. column 13 and 17). Traversals generated with the *disentanglement lib* implementation of Locatello et al. (2019).

The posterior is again Gaussian $p(\mathbf{z}_n|\mathbf{x}_n, \theta, M(\beta)) = \mathcal{N}(\mathbf{z}_n; \tilde{\mu}_{\mathbf{z}|\mathbf{x}}(\beta, n), \tilde{\Sigma}_{\mathbf{z}|\mathbf{x}}(\beta))$ where

$$\tilde{\mu}_{\mathbf{z}|\mathbf{x}}(\beta, n) = \tilde{\Sigma}^{-1}_{\mathbf{z}|\mathbf{x}}(\beta) M(\beta) W^\top D^{-1} \mathbf{x}_n$$
$$\text{and } \tilde{\Sigma}_{\mathbf{z}|\mathbf{x}}(\beta) = (M(\beta) W^\top D^{-1} W + \mathrm{I})^{-1}$$

Here we have taken care to explicitly reveal all of the direct dependencies on β.

Mean-field variational inference, $q(\mathbf{z}_n) = \prod_k q_{n,k}(z_{k,d})$, will return a diagonal Gaussian approximation to the true posterior with the same mean and matching diagonal precision,

$$q(\mathbf{z}_n|\mathbf{x}_n, \theta, M(\beta)) = \mathcal{N}\left(\mathbf{z}_n; \tilde{\mu}_{\mathbf{z}|\mathbf{x}}(\beta, n), \Sigma_q(\beta)\right) ,$$
$$\text{where } \Sigma_q^{-1}(\beta) = \mathrm{diag}\left(\tilde{\Sigma}^{-1}_{\mathbf{z}|\mathbf{x}}(\beta)\right)$$

We notice that the posterior mean is a linear combination of the observations $\tilde{\mu}_{\mathbf{z}|\mathbf{x}}(\beta, n) = R(\beta)\mathbf{x}_n$ where $R(\beta) = \tilde{\Sigma}_{\mathbf{z}|\mathbf{x}}(\beta) M(\beta) W^\top D^{-1}$ are recognition weights. Notice that the recognition weights and the posterior variances are the same for all data points: they do not depend on n. The free-energy is then

$$\mathcal{F}(q, \theta, \beta) = \mathbb{E}_{q(z)}(\log p(x|z)) - \mathrm{KL}(q(z)|p(z))$$

with the reconstruction term being

$$\begin{aligned}
&\mathbb{E}_{q(z)}(\log p(x|z)) = \\
&= -\frac{1}{2\beta}\sum_{n=1}^{N} \mathbf{x}_n^\top (D^{-1} - 2R^\top W^\top D^{-1} \\
&\quad + R^\top W^\top D^{-1} W R)\mathbf{x}_n \\
&\quad - \frac{N}{2\beta}\log\det(2\pi D) - \frac{N}{2\beta}\mathrm{trace}(W^\top D^{-1}\Sigma_q) \\
&= -\frac{N}{2\beta}\Big(\mathrm{trace}\Big((D^{-1} - 2R^\top W^\top D^{-1} \\
&\quad + R^\top W^\top D^{-1} W R)(\Sigma_{\mathbf{x}} + \mu_{\mathbf{x}}\mu_{\mathbf{x}}^\top)\Big) \\
&\quad + \log\det(2\pi D) + \mathrm{trace}(W^\top D^{-1} W \Sigma_q)\Big) \qquad (16)
\end{aligned}$$

and the KL or regularization term being

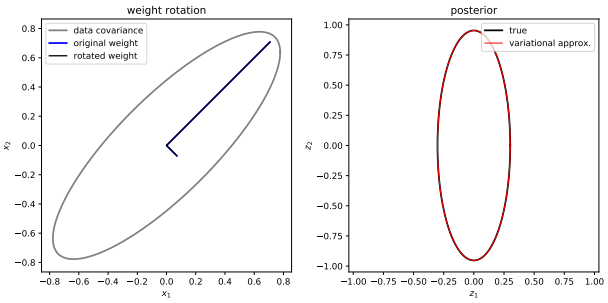
$$\begin{aligned}
&\mathrm{KL}(q(z)|p(z)) = \\
&= -\frac{NK}{2} - \frac{N}{2}\log\det(\Sigma_q) + \frac{N}{2}\mathrm{trace}(\Sigma_q) \\
&\quad + \frac{1}{2}\sum_{n=1}^{N} \mathbf{x}_n^\top R^\top R \mathbf{x}_n \\
&= -\frac{N}{2}\Big(K + \log\det(\Sigma_q) - \mathrm{trace}(\Sigma_q) \\
&\quad - \mathrm{trace}(R^\top R(\Sigma_{\mathbf{x}} + \mu_{\mathbf{x}}\mu_{\mathbf{x}}^\top))\Big).
\end{aligned}$$

We will now consider the objective functions and the posterior distributions in several cases to reason about the parameter estimates arising from the methods above.

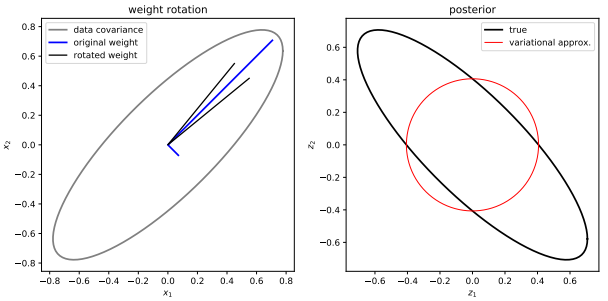
B.3 Experiment 1: mean field VI applied to factor analysis yields the PCA directions

Consider the situation where we know a maximum likelihood solution of the weights W_{ML}. For simplicity we select the solution W_{ML} which has orthogonal weights in the observation space. We then rotate this solution by an amount θ so that $W'_{\mathrm{ML}} = R(\theta)W_{\mathrm{ML}}$. The resulting weights are no longer orthogonal (assuming the rotation is not an integer multiple of $\pi/2$). We compute the log-likelihood (which will not change) and the free-energy (which will change) and plot the true and approximate posterior covariance (which does not depend on the datapoint value x_n).

First here are the weights are aligned with the true ones. The log-likelihood and the free-energy take the same value of -17.82 nats.



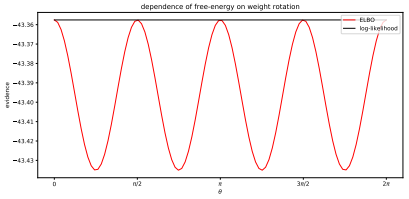
Second, here are the weights rotated $\pi/4$ and the log-likelihood is -17.82 nats and the free-energy -57.16 nats.



When varying the rotation away fom the orthogonal setting, θ, the plots above indicate that orthogonal settings of the weights ($\theta = m\pi/2$ where $m = 0, 1, 2, ...$) lead to factorized posteriors. In these cases the KL between the approximate posterior and the true posterior is zero and the free-energy is equal to the log-likelihood. This will be the optimal free-energy for any weight setting (due to the fact that it is equal to the true log-likelihood which is maximal, and the free-energy is a lower bound of this quantity.) For intermediate values of θ the posterior is correlated and the free-energy is not tight to the log likelihood.

Now let's plot the free-energy and the log-likelihood

as θ is varied. This shows that the free-energy prefers orthogonal settings of the weights as this leads to factorized posteriors, even though the log-likelihood is insensitive to θ. So, variational inference recovers the same weight directions as the PCA solution.



The above shows that the bias inherent in variational methods will cause them to break the symmetry in the log-likelihood and find orthorgonal latent components. This occurs because orthoginal components result in posterior distributions that are factorized. These are then well-modelled by the variational approximation and result in a small KL between the approximate and true posteriors.

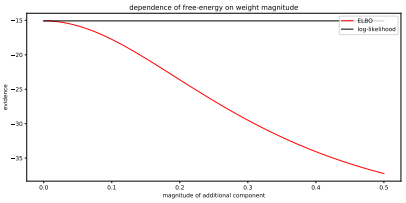
B.4 Experiment 2: mean field VI applied to over-complete factor analysis prunes out the additional latent dimensions

A similar effect occurs if we model 2D data with a 3D latent space. Many settings of the weights attain the maximum of the likelihood, including solutions which use all three latent variables. However, the optimal solution for VI is to retain two orthogonal components and to set the magnitude of the third component to zero. This solution a) returns weights that maximise the likelihood, and b) has a factorised posterior distribution (the pruned component having a posterior equal to its prior) that therefore incurs no cost $\mathrm{KL}(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x},\theta)) = 0$. In this way the bound becomes tight.

Here's an example of this effect. We consider a model of the form:

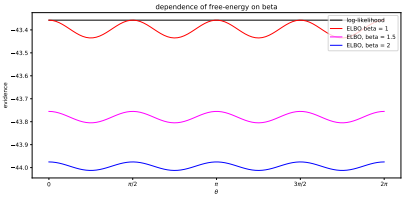
$$\mathbf{x} = \frac{\alpha}{\sqrt{2}}\begin{bmatrix}1\\1\end{bmatrix} z_1 + \frac{\beta}{\sqrt{2}}\begin{bmatrix}1\\1\end{bmatrix} z_2 + \frac{\rho}{\sqrt{2}}\begin{bmatrix}1\\-1\end{bmatrix} z_3 + \epsilon \tag{17}$$

We set $\alpha^2 + \beta^2 = 1$ so that all models imply the same covariance and set this to be the maximum likelihood covariance by construction. We then consider varying α from 0 to 1/2. The setting equal to 0 attains the maximum of the free-energy, even though it has the same likelihood as any other setting.



B.5 Experiment 3: The β-VAE also yields the PCA components, changing β has no effect on the direction of the estimated components in the FA model

How does the setting of β change things? Here we rerun experiment 1 for different values of β.



In this example, changing β in this example just reduces the amplitude of the fluctuations in the free-energy, but it does not change the directions found. A similar observation applies to the pruning experiment.

Increasing β will increase the uncertainty in the posterior as it is like reducing the number of observations (or increasing the observation noise, from the perspective of q).

B.6 Summary of Factor Analysis Experiments

The behaviours introduced by the β-VAE appear relatively benign, and perhaps even helpful, in the linear case: VI is breaking the degeneracy of the maximum likelihood solution in a sensible way: selecting amongst the maximum likelihood solutions to find those that have orthogonal components and removing spurious latent dimensions. This should be tempered by the fact that the β generalization recovered precisely the same solutions and so it was necessary to obtain the desired behaviour in the PCA case.

Similar effects will occur in deep generative models, not least since these typically also have a Gaussian prior over latent variables, and these latents are initially linearly transformed, thereby resulting in a similar degeneracy to factor analysis.

However, the behaviours above benefited from the fact that maximum-likelihood solutions could be found in which the posterior distribution over latent variables factorized. In real world examples, for example in

deep generative models, this will not be case. In such cases, these same effects will cause the variational free-energy and its β-generalization to **bias the estimated parameters far away from maximum-likelihood settings, toward those settings that imply factorized Gaussian posteriors over the latent variables**.

B.7 Independent Component Analysis

We now apply VI and the β free-energy method to ICA. We're interested the properties of the variational objective and the β-VI objective and so we 1. fit the data using the true generative model to investigate the biases in VI and β-VI 2. do not use amortized inference, just optimizing the approximating distributions for each data point (this is possible for these small examples).

The linear independent component analysis generative model we use is defined as follows. Let $\mathbf{x} \in \mathbb{R}^L$ and $\mathbf{z} \in \mathbb{R}^K$.

$$
\begin{aligned}
&\text{for } \; n = 1...N \\
&\quad \text{for } \; k = 1...K \\
&\qquad z_{n,k} \sim \text{Student-t}(0, \sigma, v), \\
&\quad \mathbf{x}_n \sim \mathcal{N}(W\mathbf{z}_n, D) \;\; \text{where } \; D = \text{diag}([\sigma_1^2, ..., \sigma_{\text{D}}^2])
\end{aligned}
$$

We apply mean-field variational inference, $q(\mathbf{z}_n) = \prod_k q_{n,k}(z_{k,d})$, and use Gaussian distributions for each factor $q_{n,k}(z_{n,k}) = \mathcal{N}(z_{n,k}; \mu_{n,k}, \sigma^2_{n,k})$.

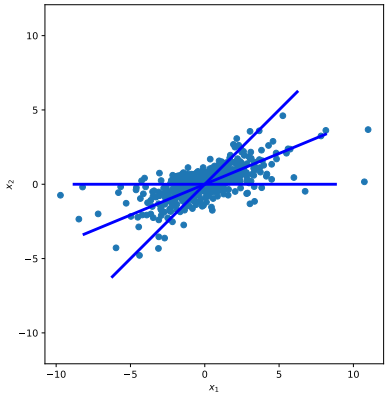
The free-energy is computed as follows: The reconstruction term is identical to PCA: an avergage of a quadratic form wrt to a Gaussian, which is analytic. The KL is broken down into the differential entropy of q which is also analytic and the cross-entropy with the prior which we evaluate by numerical integration (finite differences). There is a cross-entropy term for each latent variable which is one reason why the code is slow (requiring N 1D numerical integrations). The gradient of the free-energy wrt the parameters W and the means and variances of the Gaussian q distributions are computed using autograd.

In order to be as certain as possible that we are finding a global maximum of the free-energies, all experiments initialise at the true value of the parameters and then ensure that each gradient step improves the free-energy. Stochastic optimization or a procedure that accepted all steps regardless of the change in the objective would be faster, but they might also move us into the basis of attraction of a worse (local) optima.

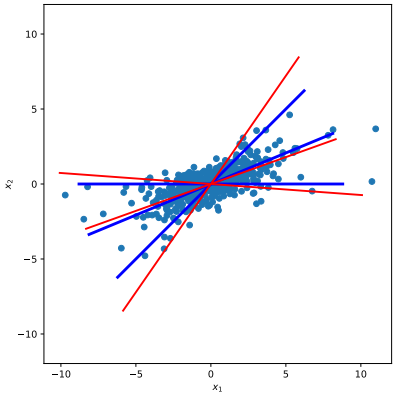
B.8 Experiment 1: Learning in over-complete ICA

Now we define the dataset. We use a very sparse Student's t-distribtion with $v = 3.5$. For $v < 4$ the the kurtosis is undefined so the model is fairly simple to estimate (it's a long way away from the degenerate factor analysis case which is recovered in the limit $v \to \infty$).

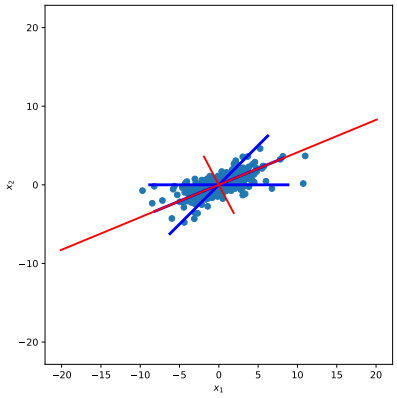
We use three latent components and a two dimensional observed space. The directions of the three weights are shown in blue below with data as blue circles.



First we run variational inference finding components (shown in red below) which are more orthogonal than the true directions. This bias is in this directions as this reduces the dependencies (explaining away) in the underlying posterior.



Second we run β-VI with $\beta = 5$. Two components are now found that are orthogonal with one component pruned from the solution.



In this case the bias is so great that the true component directions are not discovered. Instead the components are forced into the orthogonal setting regardless of the structure in the data.

B.9 Summary of Independent Component Analysis experiment

The ICA example illustrates that this approach – of relying on a bias inherent in VI to discover meaningful components – will sometimes return meaningful structure (e.g. in the PCA experiments above). However it does not seem to be a sensible way of doing so in general. For example, explaining away often means that the true components will be entangled in the posterior, as is the case in the ICA example, and the variational bias will then move us away from this solution. The β-VI generalisation only enhances this undesirable bias.