Probabilistic & Unsupervised Learning

Expectation Propagation

Maneesh Sahani
maneesh@gatsby.ucl.ac.uk

Gatsby Computational Neuroscience Unit, and
MSc ML/CSML, Dept Computer Science
University College London

Term 1, Autumn 2013

Intractabilities and approximations

- Inference – computational intractability
  - Factored variational approx
  - Loopy BP/EP/Power EP
  - Gibbs sampling, other MCMC
- Inference – analytic intractability
  - Laplace approximation (global)
  - Parametric variational approx (for special cases).
  - Message approximations (linearised, sigma-point, Laplace)
  - Assumed-density methods and Expectation-Propagation
  - (Sequential) Monte-Carlo methods
- Learning – intractable partition function
  - Constrastive divergence
  - Sampling parameters
  - Score-matching
- Model selection
  - Laplace approximation / BIC
  - Variational Bayes
  - (Annealed) importance sampling
  - Reversible jump MCMC

Not a complete list!

Intractabilities and approximations

- Inference – computational intractability
  - Factored variational approx
  - Loopy BP/EP/Power EP
  - Gibbs sampling, other MCMC
- Inference – analytic intractability
  - Laplace approximation (global)
  - Parametric variational approx (for special cases).
  - Message approximations (linearised, sigma-point, Laplace)
  - Assumed-density methods and Expectation-Propagation
  - (Sequential) Monte-Carlo methods
- Learning – intractable partition function
  - Constrastive divergence
  - Sampling parameters
  - Score-matching
- Model selection
  - Laplace approximation / BIC
  - Variational Bayes
  - (Annealed) importance sampling
  - Reversible jump MCMC

Not a complete list!

Intractabilities and approximations

- Inference – computational intractability
  - Factored variational approx
  - Loopy BP/EP/Power EP
  - Gibbs sampling, other MCMC
- Inference – analytic intractability
  - Laplace approximation (global)
  - Parametric variational approx (for special cases).
  - Message approximations (linearised, sigma-point, Laplace)
  - Assumed-density methods and Expectation-Propagation
  - (Sequential) Monte-Carlo methods
- Learning – intractable partition function
  - Constrastive divergence
  - Sampling parameters
  - Score-matching
- Model selection
  - Laplace approximation / BIC
  - Variational Bayes
  - (Annealed) importance sampling
  - Reversible jump MCMC

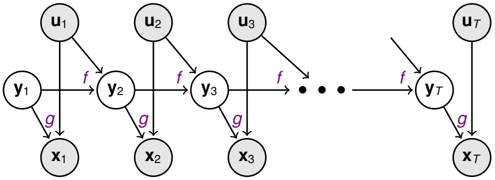
Not a complete list!

Intractabilities and approximations

- Inference – computational intractability
  - Factored variational approx
  - Loopy BP/EP/Power EP
  - Gibbs sampling, other MCMC
- Inference – analytic intractability
  - Laplace approximation (global)
  - Parametric variational approx (for special cases).
  - Message approximations (linearised, sigma-point, Laplace)
  - Assumed-density methods and Expectation-Propagation
  - (Sequential) Monte-Carlo methods
- Learning – intractable partition function
  - Constrastive divergence
  - Sampling parameters
  - Score-matching
- Model selection
  - Laplace approximation / BIC
  - Variational Bayes
  - (Annealed) importance sampling
  - Reversible jump MCMC

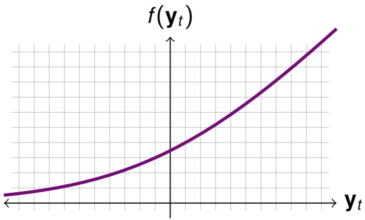
Not a complete list!

Nonlinear state-space model (NLSSM)

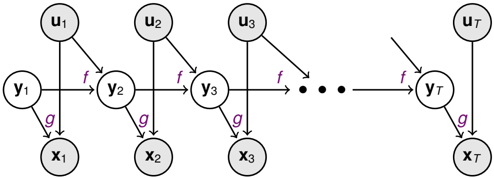


$$\mathbf{y}_{t+1} = f(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{w}_t$$
$$\mathbf{x}_t = g(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{v}_t$$

$\mathbf{w}_t, \mathbf{v}_t$ usually still Gaussian.



Nonlinear state-space model (NLSSM)



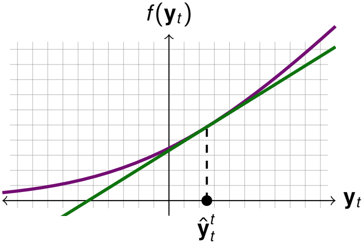
$$\mathbf{y}_{t+1} = f(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{w}_t$$
$$\mathbf{x}_t = g(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{v}_t$$

$\mathbf{w}_t, \mathbf{v}_t$ usually still Gaussian.

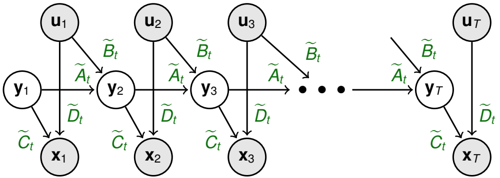
Extended Kalman Filter (EKF): linearise nonlinear functions about current estimate, $\hat{\mathbf{y}}_t^t$:

$$\mathbf{y}_{t+1} \approx f(\hat{\mathbf{y}}_t^t, \mathbf{u}_t) + \left.\frac{\partial f}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{w}_t$$

$$\mathbf{x}_t \approx g(\hat{\mathbf{y}}_t^t, \mathbf{u}_t) + \left.\frac{\partial g}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{v}_t$$



Nonlinear state-space model (NLSSM)



$$\mathbf{y}_{t+1} = f(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{w}_t$$

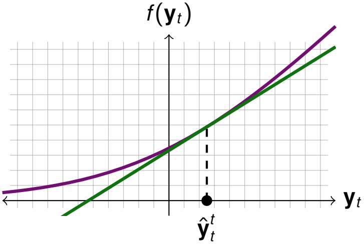
$$\mathbf{x}_t = g(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{v}_t$$

$\mathbf{w}_t, \mathbf{v}_t$ usually still Gaussian.

Extended Kalman Filter (EKF): linearise nonlinear functions about current estimate, $\hat{\mathbf{y}}_t^t$:

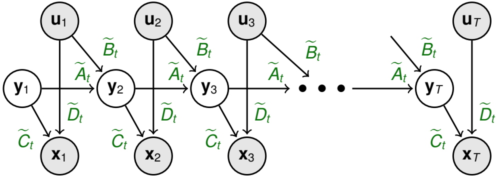
$$\mathbf{y}_{t+1} \approx \underbrace{f(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{B}_t \mathbf{u}_t} + \underbrace{\left.\frac{\partial f}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{A}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{w}_t$$

$$\mathbf{x}_t \approx \underbrace{g(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{D}_t \mathbf{u}_t} + \underbrace{\left.\frac{\partial g}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{C}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{v}_t$$



Run the Kalman filter (smoother) on non-stationary linearised system $(\widetilde{A}_t, \widetilde{B}_t, \widetilde{C}_t, \widetilde{D}_t)$:

Nonlinear state-space model (NLSSM)



$$\mathbf{y}_{t+1} = f(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{w}_t$$

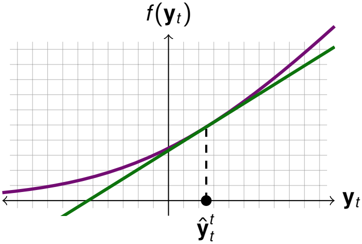
$$\mathbf{x}_t = g(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{v}_t$$

$\mathbf{w}_t, \mathbf{v}_t$ usually still Gaussian.

Extended Kalman Filter (EKF): linearise nonlinear functions about current estimate, $\hat{\mathbf{y}}_t^t$:

$$\mathbf{y}_{t+1} \approx \underbrace{f(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{B}_t \mathbf{u}_t} + \underbrace{\left.\frac{\partial f}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{A}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{w}_t$$

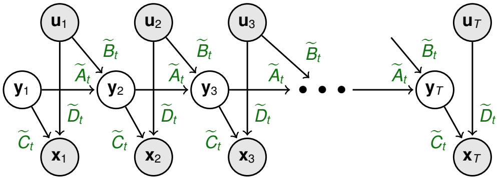
$$\mathbf{x}_t \approx \underbrace{g(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{D}_t \mathbf{u}_t} + \underbrace{\left.\frac{\partial g}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{C}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{v}_t$$



Run the Kalman filter (smoother) on non-stationary linearised system $(\widetilde{A}_t, \widetilde{B}_t, \widetilde{C}_t, \widetilde{D}_t)$:

- Adaptively approximates non-Gaussian messages by Gaussians.

Nonlinear state-space model (NLSSM)



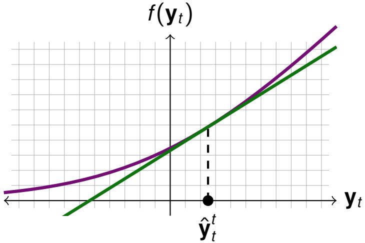
$$\mathbf{y}_{t+1} = f(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{w}_t$$
$$\mathbf{x}_t = g(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{v}_t$$

$\mathbf{w}_t, \mathbf{v}_t$ usually still Gaussian.

Extended Kalman Filter (EKF): linearise nonlinear functions about current estimate, $\hat{\mathbf{y}}_t^t$:

$$\mathbf{y}_{t+1} \approx \underbrace{f(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{B}_t \mathbf{u}_t} + \underbrace{\frac{\partial f}{\partial \mathbf{y}_t}\bigg|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{A}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{w}_t$$

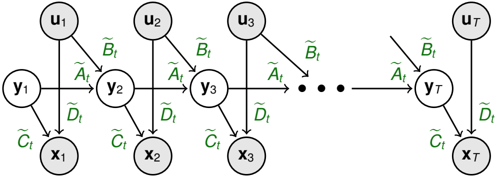
$$\mathbf{x}_t \approx \underbrace{g(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{D}_t \mathbf{u}_t} + \underbrace{\frac{\partial g}{\partial \mathbf{y}_t}\bigg|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{C}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{v}_t$$



Run the Kalman filter (smoother) on non-stationary linearised system $(\widetilde{A}_t, \widetilde{B}_t, \widetilde{C}_t, \widetilde{D}_t)$:

- Adaptively approximates non-Gaussian messages by Gaussians.
- Local linearisation depends on central point of distribution $\Rightarrow$ approximation degrades with increased state uncertainty.

Nonlinear state-space model (NLSSM)



$$\mathbf{y}_{t+1} = f(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{w}_t$$

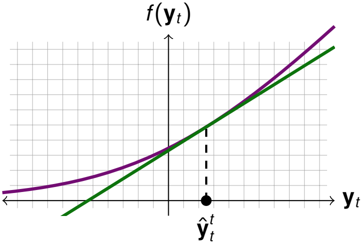
$$\mathbf{x}_t = g(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{v}_t$$

$\mathbf{w}_t, \mathbf{v}_t$ usually still Gaussian.

Extended Kalman Filter (EKF): linearise nonlinear functions about current estimate, $\hat{\mathbf{y}}_t^t$:

$$\mathbf{y}_{t+1} \approx \underbrace{f(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{B}_t \mathbf{u}_t} + \underbrace{\left.\frac{\partial f}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{A}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{w}_t$$

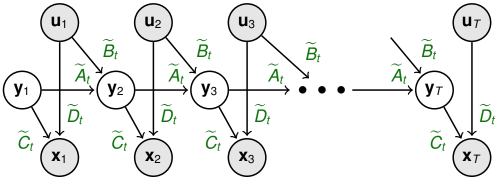
$$\mathbf{x}_t \approx \underbrace{g(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{D}_t \mathbf{u}_t} + \underbrace{\left.\frac{\partial g}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{C}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{v}_t$$



Run the Kalman filter (smoother) on non-stationary linearised system $(\widetilde{A}_t, \widetilde{B}_t, \widetilde{C}_t, \widetilde{D}_t)$:

- Adaptively approximates non-Gaussian messages by Gaussians.
- Local linearisation depends on central point of distribution $\Rightarrow$ approximation degrades with increased state uncertainty. May work acceptably for close-to-linear systems.

Nonlinear state-space model (NLSSM)



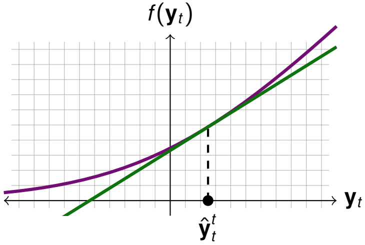
$$\mathbf{y}_{t+1} = f(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{w}_t$$
$$\mathbf{x}_t = g(\mathbf{y}_t, \mathbf{u}_t) + \mathbf{v}_t$$

$\mathbf{w}_t, \mathbf{v}_t$ usually still Gaussian.

Extended Kalman Filter (EKF): linearise nonlinear functions about current estimate, $\hat{\mathbf{y}}_t^t$:

$$\mathbf{y}_{t+1} \approx \underbrace{f(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{B}_t \mathbf{u}_t} + \underbrace{\left.\frac{\partial f}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{A}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{w}_t$$

$$\mathbf{x}_t \approx \underbrace{g(\hat{\mathbf{y}}_t^t, \mathbf{u}_t)}_{\widetilde{D}_t \mathbf{u}_t} + \underbrace{\left.\frac{\partial g}{\partial \mathbf{y}_t}\right|_{\hat{\mathbf{y}}_t^t}}_{\widetilde{C}_t} (\mathbf{y}_t - \hat{\mathbf{y}}_t^t) + \mathbf{v}_t$$



Run the Kalman filter (smoother) on non-stationary linearised system $(\widetilde{A}_t, \widetilde{B}_t, \widetilde{C}_t, \widetilde{D}_t)$:

- Adaptively approximates non-Gaussian messages by Gaussians.
- Local linearisation depends on central point of distribution $\Rightarrow$ approximation degrades with increased state uncertainty. May work acceptably for close-to-linear systems.

Can base EM-like algorithm on EKF/EKS (or alternatives).

Online learning with NLSSM message passing

Nonlinear message passing can also be used to implement online parameter learning in (non)linear latent state-space systems:

Online learning with NLSSM message passing

Nonlinear message passing can also be used to implement online parameter learning in (non)linear latent state-space systems:

Eg: for linear model, augment state vector to include the model parameters and introduce nonlinear transition $\bar{\bar{f}}$:

$$\bar{\bar{\mathbf{y}}}_t = \begin{bmatrix} \mathbf{y}_t \\ A \\ C \end{bmatrix}$$

$$\bar{\bar{\mathbf{y}}}_{t+1} = \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}_t) + \begin{bmatrix} \mathbf{w}_t \\ 0 \\ 0 \end{bmatrix} \qquad \text{with } \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}) = \bar{\bar{f}}\left(\begin{bmatrix} \mathbf{y} \\ A \\ C \end{bmatrix} \right) = \begin{bmatrix} A\mathbf{y} \\ A \\ C \end{bmatrix}$$

Online learning with NLSSM message passing

Nonlinear message passing can also be used to implement online parameter learning in (non)linear latent state-space systems:

Eg: for linear model, augment state vector to include the model parameters and introduce nonlinear transition $\bar{\bar{f}}$:

$$\bar{\bar{\mathbf{y}}}_t = \begin{bmatrix} \mathbf{y}_t \\ A \\ C \end{bmatrix}$$

$$\bar{\bar{\mathbf{y}}}_{t+1} = \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}_t) + \begin{bmatrix} \mathbf{w}_t \\ 0 \\ 0 \end{bmatrix} \qquad \text{with } \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}) = \bar{\bar{f}}\left(\begin{bmatrix} \mathbf{y} \\ A \\ C \end{bmatrix} \right) = \begin{bmatrix} A\mathbf{y} \\ A \\ C \end{bmatrix}$$

Use EKF or alternative to compute online estimates of $E[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \ldots, \mathbf{x}_t]$ and $\text{Cov}[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \ldots, \mathbf{x}_t]$. These now include mean and posterior variance of parameter estimates.

Online learning with NLSSM message passing

Nonlinear message passing can also be used to implement online parameter learning in (non)linear latent state-space systems:

Eg: for linear model, augment state vector to include the model parameters and introduce nonlinear transition $\bar{\bar{f}}$:

$$\bar{\bar{\mathbf{y}}}_t = \begin{bmatrix} \mathbf{y}_t \\ A \\ C \end{bmatrix}$$

$$\bar{\bar{\mathbf{y}}}_{t+1} = \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}_t) + \begin{bmatrix} \mathbf{w}_t \\ 0 \\ 0 \end{bmatrix} \qquad \text{with } \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}) = \bar{\bar{f}}\left(\begin{bmatrix} \mathbf{y} \\ A \\ C \end{bmatrix}\right) = \begin{bmatrix} A\mathbf{y} \\ A \\ C \end{bmatrix}$$

Use EKF or alternative to compute online estimates of $E[\bar{\bar{\mathbf{y}}}_t | \mathbf{x}_1, \ldots, \mathbf{x}_t]$ and $\mathrm{Cov}[\bar{\bar{\mathbf{y}}}_t | \mathbf{x}_1, \ldots, \mathbf{x}_t]$. These now include mean and posterior variance of parameter estimates.

- Pseudo-Bayesian approach: gives Gaussian distributions over parameters.

Online learning with NLSSM message passing

Nonlinear message passing can also be used to implement online parameter learning in (non)linear latent state-space systems:

Eg: for linear model, augment state vector to include the model parameters and introduce nonlinear transition $\bar{\bar{f}}$:

$$\bar{\bar{\mathbf{y}}}_t = \begin{bmatrix} \mathbf{y}_t \\ A \\ C \end{bmatrix}$$

$$\bar{\bar{\mathbf{y}}}_{t+1} = \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}_t) + \begin{bmatrix} \mathbf{w}_t \\ 0 \\ 0 \end{bmatrix} \qquad \text{with } \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}) = \bar{\bar{f}}\left(\begin{bmatrix} \mathbf{y} \\ A \\ C \end{bmatrix}\right) = \begin{bmatrix} A\mathbf{y} \\ A \\ C \end{bmatrix}$$

Use EKF or alternative to compute online estimates of $E[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \dots, \mathbf{x}_t]$ and $\text{Cov}[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \dots, \mathbf{x}_t]$. These now include mean and posterior variance of parameter estimates.

- Pseudo-Bayesian approach: gives Gaussian distributions over parameters.
- Can model nonstationarity by assuming non-zero innovations noise in A, C.

Online learning with NLSSM message passing

Nonlinear message passing can also be used to implement online parameter learning in (non)linear latent state-space systems:

Eg: for linear model, augment state vector to include the model parameters and introduce nonlinear transition $\bar{\bar{f}}$:

$$\bar{\bar{\mathbf{y}}}_t = \begin{bmatrix} \mathbf{y}_t \\ A \\ C \end{bmatrix}$$

$$\bar{\bar{\mathbf{y}}}_{t+1} = \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}_t) + \begin{bmatrix} \mathbf{w}_t \\ 0 \\ 0 \end{bmatrix} \qquad \text{with } \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}) = \bar{\bar{f}}\left(\begin{bmatrix} \mathbf{y} \\ A \\ C \end{bmatrix}\right) = \begin{bmatrix} A\mathbf{y} \\ A \\ C \end{bmatrix}$$

Use EKF or alternative to compute online estimates of $E[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \ldots, \mathbf{x}_t]$ and $\mathrm{Cov}[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \ldots, \mathbf{x}_t]$. These now include mean and posterior variance of parameter estimates.

- Pseudo-Bayesian approach: gives Gaussian distributions over parameters.
- Can model nonstationarity by assuming non-zero innovations noise in A, C.
- Not simple to implement for Q and R (e.g. covariance constraints?).

Online learning with NLSSM message passing

Nonlinear message passing can also be used to implement online parameter learning in (non)linear latent state-space systems:

Eg: for linear model, augment state vector to include the model parameters and introduce nonlinear transition $\bar{\bar{f}}$:

$$\bar{\bar{\mathbf{y}}}_t = \begin{bmatrix} \mathbf{y}_t \\ A \\ C \end{bmatrix}$$

$$\bar{\bar{\mathbf{y}}}_{t+1} = \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}_t) + \begin{bmatrix} \mathbf{w}_t \\ 0 \\ 0 \end{bmatrix} \qquad \text{with } \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}) = \bar{\bar{f}}\left(\begin{bmatrix} \mathbf{y} \\ A \\ C \end{bmatrix}\right) = \begin{bmatrix} A\mathbf{y} \\ A \\ C \end{bmatrix}$$

Use EKF or alternative to compute online estimates of $E[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \ldots, \mathbf{x}_t]$ and $\text{Cov}[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \ldots, \mathbf{x}_t]$. These now include mean and posterior variance of parameter estimates.

- Pseudo-Bayesian approach: gives Gaussian distributions over parameters.
- Can model nonstationarity by assuming non-zero innovations noise in A, C.
- Not simple to implement for Q and R (e.g. covariance constraints?).
- May be faster than EM/gradient approaches.

Online learning with NLSSM message passing

Nonlinear message passing can also be used to implement online parameter learning in (non)linear latent state-space systems:

Eg: for linear model, augment state vector to include the model parameters and introduce nonlinear transition $\bar{\bar{f}}$:

$$\bar{\bar{\mathbf{y}}}_t = \begin{bmatrix} \mathbf{y}_t \\ A \\ C \end{bmatrix}$$

$$\bar{\bar{\mathbf{y}}}_{t+1} = \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}_t) + \begin{bmatrix} \mathbf{w}_t \\ 0 \\ 0 \end{bmatrix} \qquad \text{with } \bar{\bar{f}}(\bar{\bar{\mathbf{y}}}) = \bar{\bar{f}}\left(\begin{bmatrix} \mathbf{y} \\ A \\ C \end{bmatrix}\right) = \begin{bmatrix} A\mathbf{y} \\ A \\ C \end{bmatrix}$$

Use EKF or alternative to compute online estimates of $E[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \ldots, \mathbf{x}_t]$ and $\mathrm{Cov}[\bar{\bar{\mathbf{y}}}_t|\mathbf{x}_1, \ldots, \mathbf{x}_t]$. These now include mean and posterior variance of parameter estimates.

- Pseudo-Bayesian approach: gives Gaussian distributions over parameters.
- Can model nonstationarity by assuming non-zero innovations noise in A, C.
- Not simple to implement for Q and R (e.g. covariance constraints?).
- May be faster than EM/gradient approaches.

Sometimes called the joint-EKF approach.

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z} P(x_t|y_t) \int dy_{t-1}\, P(y_t|y_{t-1}) P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z} P(x_t|y_t) \int dy_{t-1} \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}),\, Q)} \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1},\, V_{t-1})}$$

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z}P(x_t|y_t)\int dy_{t-1}\, P(y_t|y_{t-1})P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z}P(x_t|y_t)\int dy_{t-1} \quad \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}),\, Q)} \quad \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1},\, V_{t-1})}$$

- Linearisation at the peak (EKF) is only one approach.

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z} P(x_t|y_t) \int dy_{t-1}\, P(y_t|y_{t-1}) P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z} P(x_t|y_t) \int dy_{t-1} \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}),\, Q)} \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1},\, V_{t-1})}$$

- Linearisation at the peak (EKF) is only one approach.
- Laplace filter: use mode and curvature of integrand.

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z} P(x_t|y_t) \int dy_{t-1}\, P(y_t|y_{t-1}) P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z} P(x_t|y_t) \int dy_{t-1} \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}), Q)} \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1}, V_{t-1})}$$

- Linearisation at the peak (EKF) is only one approach.
- Laplace filter: use mode and curvature of integrand.
- Sigma-point (“unscented”) filter:

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z} P(x_t|y_t) \int dy_{t-1}\, P(y_t|y_{t-1}) P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z} P(x_t|y_t) \int dy_{t-1} \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}), Q)} \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1}, V_{t-1})}$$

- Linearisation at the peak (EKF) is only one approach.
- Laplace filter: use mode and curvature of integrand.
- Sigma-point ("unscented") filter:
  - Evaluate $f(\hat{\mathbf{y}}_{t-1}), f(\hat{\mathbf{y}}_{t-1} \pm \boldsymbol{\lambda}(V)_{t-1})$ for eigenvectors $\boldsymbol{\lambda}$.

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z} P(x_t|y_t) \int dy_{t-1}\, P(y_t|y_{t-1}) P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z} P(x_t|y_t) \int dy_{t-1} \quad \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}), Q)} \quad \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1}, V_{t-1})}$$

- Linearisation at the peak (EKF) is only one approach.
- Laplace filter: use mode and curvature of integrand.
- Sigma-point ("unscented") filter:
  - Evaluate $f(\hat{\mathbf{y}}_{t-1}), f(\hat{\mathbf{y}}_{t-1} \pm \boldsymbol{\lambda}(V)_{t-1})$ for eigenvectors $\boldsymbol{\lambda}$.
  - "Fit" Gaussian to these $2K + 1$ points.

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z} P(x_t|y_t) \int dy_{t-1}\, P(y_t|y_{t-1}) P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z} P(x_t|y_t) \int dy_{t-1} \quad \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}),\, Q)} \quad \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1},\, V_{t-1})}$$

- Linearisation at the peak (EKF) is only one approach.
- Laplace filter: use mode and curvature of integrand.
- Sigma-point ("unscented") filter:
  - Evaluate $f(\hat{\mathbf{y}}_{t-1}), f(\hat{\mathbf{y}}_{t-1} \pm \boldsymbol{\lambda}(V)_{t-1})$ for eigenvectors $\boldsymbol{\lambda}$.
  - "Fit" Gaussian to these $2K+1$ points.
  - Equivalent to numerical evaluation of mean and covariance by Gaussian quadrature.

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z} P(x_t|y_t) \int dy_{t-1}\, P(y_t|y_{t-1}) P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z} P(x_t|y_t) \int dy_{t-1} \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}), Q)} \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1}, V_{t-1})}$$

- Linearisation at the peak (EKF) is only one approach.
- Laplace filter: use mode and curvature of integrand.
- Sigma-point ("unscented") filter:
  - Evaluate $f(\hat{\mathbf{y}}_{t-1}), f(\hat{\mathbf{y}}_{t-1} \pm \boldsymbol{\lambda}(V)_{t-1})$ for eigenvectors $\boldsymbol{\lambda}$.
  - "Fit" Gaussian to these $2K + 1$ points.
  - Equivalent to numerical evaluation of mean and covariance by Gaussian quadrature.
  - One form of "Assumed Density Filtering" and EP.

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z} P(x_t|y_t) \int dy_{t-1}\, P(y_t|y_{t-1}) P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z} P(x_t|y_t) \int dy_{t-1} \quad \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}),\, Q)} \quad \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1},\, V_{t-1})}$$

- Linearisation at the peak (EKF) is only one approach.
- Laplace filter: use mode and curvature of integrand.
- Sigma-point ("unscented") filter:
  - Evaluate $f(\hat{\mathbf{y}}_{t-1}), f(\hat{\mathbf{y}}_{t-1} \pm \boldsymbol{\lambda}(V)_{t-1})$ for eigenvectors $\boldsymbol{\lambda}$.
  - "Fit" Gaussian to these $2K + 1$ points.
  - Equivalent to numerical evaluation of mean and covariance by Gaussian quadrature.
  - One form of "Assumed Density Filtering" and EP.
- Parametric variational: argmin $\mathbf{KL}\left[\mathcal{N}\left(\hat{\mathbf{y}}_t, \hat{V}_t\right) \middle\| \int dy_{t-1} \ldots\right]$. Requires Gaussian expectations of $\log \int \Rightarrow$ may be challenging.

Other message approximations

Consider the forward messages on a latent chain:

$$P(y_t|x_{1:t}) = \frac{1}{Z} P(x_t|y_t) \int dy_{t-1}\, P(y_t|y_{t-1}) P(y_{t-1}|x_{1:t-1})$$

We want to approximate the messages to retain a tractable form (i.e. Gaussian).

$$\tilde{P}(y_t|x_{1:t}) \approx \frac{1}{Z} P(x_t|y_t) \int dy_{t-1} \quad \underbrace{P(y_t|y_{t-1})}_{\mathcal{N}(f(\mathbf{y}_{t-1}), Q)} \quad \underbrace{\tilde{P}(y_{t-1}|x_{1:t-1})}_{\mathcal{N}(\hat{\mathbf{y}}_{t-1}, V_{t-1})}$$

- Linearisation at the peak (EKF) is only one approach.
- Laplace filter: use mode and curvature of integrand.
- Sigma-point ("unscented") filter:
  - Evaluate $f(\hat{\mathbf{y}}_{t-1})$, $f(\hat{\mathbf{y}}_{t-1} \pm \boldsymbol{\lambda}(V)_{t-1})$ for eigenvectors $\boldsymbol{\lambda}$.
  - "Fit" Gaussian to these $2K + 1$ points.
  - Equivalent to numerical evaluation of mean and covariance by Gaussian quadrature.
  - One form of "Assumed Density Filtering" and EP.
- Parametric variational: argmin $\mathbf{KL}\left[\mathcal{N}(\hat{\mathbf{y}}_t, \hat{V}_t) \middle\| \int dy_{t-1} \; \ldots\right]$. Requires Gaussian expectations of log $\int \Rightarrow$ may be challenging.
- The other KL: argmin $\mathbf{KL}\left[\int dy_{t-1} \middle\| \mathcal{N}(\hat{\mathbf{y}}_t, \hat{V}_t)\right]$ needs only first and second moments of nonlinear message $\Rightarrow$ EP.

Variational learning

Free energy:

$$\mathcal{F}(q,\theta) = \langle \log P(\mathcal{X},\mathcal{Y}|\theta) \rangle_{q(\mathcal{Y}|\mathcal{X})} + \mathbf{H}[q] = \log P(\mathcal{X}|\theta) - \mathbf{KL}[q(\mathcal{Y}) \| P(\mathcal{Y}|\mathcal{X},\theta)] \le \ell(\theta)$$

Variational learning

Free energy:

$$\mathcal{F}(q,\theta) = \langle \log P(\mathcal{X},\mathcal{Y}|\theta)\rangle_{q(\mathcal{Y}|\mathcal{X})} + \mathbf{H}[q] = \log P(\mathcal{X}|\theta) - \mathbf{KL}[q(\mathcal{Y})\|P(\mathcal{Y}|\mathcal{X},\theta)] \leq \ell(\theta)$$

E-steps:

- Exact EM: $q(\mathcal{Y}) = \operatorname*{argmax}_q \mathcal{F} = P(\mathcal{Y}|\mathcal{X},\theta)$

Variational learning

Free energy:

$$\mathcal{F}(q,\theta) = \langle \log P(\mathcal{X},\mathcal{Y}|\theta)\rangle_{q(\mathcal{Y}|\mathcal{X})} + \mathbf{H}[q] = \log P(\mathcal{X}|\theta) - \mathbf{KL}[q(\mathcal{Y})\|P(\mathcal{Y}|\mathcal{X},\theta)] \leq \ell(\theta)$$

E-steps:

- Exact EM: $q(\mathcal{Y}) = \underset{q}{\operatorname{argmax}}\, \mathcal{F} = P(\mathcal{Y}|\mathcal{X},\theta)$
  - Saturates bound: converges to local maximum of likelihood.

Variational learning

Free energy:

$$\mathcal{F}(q,\theta) = \langle \log P(\mathcal{X},\mathcal{Y}|\theta)\rangle_{q(\mathcal{Y}|\mathcal{X})} + \mathbf{H}[q] = \log P(\mathcal{X}|\theta) - \mathbf{KL}[q(\mathcal{Y})\|P(\mathcal{Y}|\mathcal{X},\theta)] \leq \ell(\theta)$$

E-steps:

- Exact EM: $q(\mathcal{Y}) = \operatorname{argmax}_q \mathcal{F} = P(\mathcal{Y}|\mathcal{X},\theta)$
  - Saturates bound: converges to local maximum of likelihood.
- (Factored) variational approximation:

$$q(\mathcal{Y}) = \operatorname*{argmax}_{q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)} \mathcal{F} = \operatorname*{argmin}_{q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)} \mathbf{KL}[q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)\|P(\mathcal{Y}|\mathcal{X},\theta)]$$

Variational learning

Free energy:

$$\mathcal{F}(q, \theta) = \langle \log P(\mathcal{X}, \mathcal{Y}|\theta) \rangle_{q(\mathcal{Y}|\mathcal{X})} + \mathbf{H}[q] = \log P(\mathcal{X}|\theta) - \mathbf{KL}[q(\mathcal{Y}) \| P(\mathcal{Y}|\mathcal{X}, \theta)] \leq \ell(\theta)$$

E-steps:

- Exact EM: $q(\mathcal{Y}) = \underset{q}{\operatorname{argmax}} \, \mathcal{F} = P(\mathcal{Y}|\mathcal{X}, \theta)$
  - Saturates bound: converges to local maximum of likelihood.

- (Factored) variational approximation:

  $$q(\mathcal{Y}) = \underset{q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)}{\operatorname{argmax}} \mathcal{F} = \underset{q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)}{\operatorname{argmin}} \mathbf{KL}[q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2) \| P(\mathcal{Y}|\mathcal{X}, \theta)]$$

  - Increases bound: converges, but not necessarily to ML.

Variational learning

Free energy:

$$\mathcal{F}(q,\theta) = \langle \log P(\mathcal{X},\mathcal{Y}|\theta)\rangle_{q(\mathcal{Y}|\mathcal{X})} + \mathbf{H}[q] = \log P(\mathcal{X}|\theta) - \mathbf{KL}[q(\mathcal{Y})\|P(\mathcal{Y}|\mathcal{X},\theta)] \leq \ell(\theta)$$

E-steps:

- Exact EM: $q(\mathcal{Y}) = \underset{q}{\operatorname{argmax}}\, \mathcal{F} = P(\mathcal{Y}|\mathcal{X},\theta)$
  - Saturates bound: converges to local maximum of likelihood.

- (Factored) variational approximation:

  $$q(\mathcal{Y}) = \underset{q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)}{\operatorname{argmax}} \mathcal{F} = \underset{q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)}{\operatorname{argmin}} \mathbf{KL}[q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)\|P(\mathcal{Y}|\mathcal{X},\theta)]$$

  - Increases bound: converges, but not necessarily to ML.

- Other approximations: $q(\mathcal{Y}) \approx P(\mathcal{Y}|\mathcal{X},\theta)$

Variational learning

Free energy:

$$\mathcal{F}(q,\theta) = \langle \log P(\mathcal{X},\mathcal{Y}|\theta)\rangle_{q(\mathcal{Y}|\mathcal{X})} + \mathbf{H}[q] = \log P(\mathcal{X}|\theta) - \mathbf{KL}[q(\mathcal{Y})\|P(\mathcal{Y}|\mathcal{X},\theta)] \leq \ell(\theta)$$

E-steps:

- Exact EM: $q(\mathcal{Y}) = \underset{q}{\operatorname{argmax}}\, \mathcal{F} = P(\mathcal{Y}|\mathcal{X},\theta)$
  - Saturates bound: converges to local maximum of likelihood.
- (Factored) variational approximation:

  $$q(\mathcal{Y}) = \underset{q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)}{\operatorname{argmax}} \mathcal{F} = \underset{q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)}{\operatorname{argmin}} \mathbf{KL}[q_1(\mathcal{Y}_1)q_2(\mathcal{Y}_2)\|P(\mathcal{Y}|\mathcal{X},\theta)]$$
  - Increases bound: converges, but not necessarily to ML.
- Other approximations: $q(\mathcal{Y}) \approx P(\mathcal{Y}|\mathcal{X},\theta)$
  - Usually no guarantees, but if learning converges if is frequently more accurate than the factored approximation

Approximating the posterior

Linearisation (or local Laplace, sigma-point and other such approaches) seem *ad hoc*. A more principled approach might look for an approximate q that is closest to P in some sense.

$$q = \operatorname*{argmin}_{q \in \mathcal{Q}} D(P \leftrightarrow q)$$

Approximating the posterior

Linearisation (or local Laplace, sigma-point and other such approaches) seem *ad hoc*. A more principled approach might look for an approximate q that is closest to P in some sense.

$$q = \underset{q \in \mathcal{Q}}{\operatorname{argmin}}\, D(P \leftrightarrow q)$$

Open choices:

- form of the metric D
- nature of the constraint space $\mathcal{Q}$

Approximating the posterior

Linearisation (or local Laplace, sigma-point and other such approaches) seem *ad hoc*. A more principled approach might look for an approximate q that is closest to P in some sense.

$$q = \underset{q \in \mathcal{Q}}{\operatorname{argmin}}\, D(P \leftrightarrow q)$$

Open choices:

- form of the metric D
- nature of the constraint space $\mathcal{Q}$

- Variational methods: $D = \mathbf{KL}[q \| P]$.

Approximating the posterior

Linearisation (or local Laplace, sigma-point and other such approaches) seem *ad hoc*. A more principled approach might look for an approximate q that is closest to P in some sense.

$$q = \underset{q \in \mathcal{Q}}{\operatorname{argmin}} D(P \leftrightarrow q)$$

Open choices:

- form of the metric D
- nature of the constraint space $\mathcal{Q}$

- Variational methods: $D = \mathbf{KL}[q \| P]$.
  - Choosing $\mathcal{Q}$ = $\{$tree-factored distributions$\}$ leads to efficient message passing.

Approximating the posterior

Linearisation (or local Laplace, sigma-point and other such approaches) seem *ad hoc*. A more principled approach might look for an approximate q that is closest to P in some sense.

$$q = \underset{q \in \mathcal{Q}}{\operatorname{argmin}}\, D(P \leftrightarrow q)$$

Open choices:

- form of the metric D
- nature of the constraint space $\mathcal{Q}$

- Variational methods: $D = \mathbf{KL}[q \| P]$.
  - Choosing $\mathcal{Q}$ = {tree-factored distributions} leads to efficient message passing.
- Can we use other divergences?

The other KL

What about the ‘other’ KL ($q = \operatorname{argmin} \mathbf{KL}[P \| q]$)?

The other KL

What about the ‘other’ KL ($q = \operatorname{argmin} \mathbf{KL}[P \| q]$)?

For a factored approximation the (clique) marginals obtained by minimising this KL are correct:

The other KL

What about the 'other' KL ($q = \operatorname{argmin} \mathbf{KL}[P \| q]$)?

For a factored approximation the (clique) marginals obtained by minimising this KL are correct:

$$\operatorname*{argmin}_{q_i} \mathbf{KL}\Big[P(\mathcal{Y}|\mathcal{X}) \Big\| \prod q_j(\mathcal{Y}_j|\mathcal{X})\Big] = \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}\ P(\mathcal{Y}|\mathcal{X}) \log \prod_j q_j(\mathcal{Y}_j|\mathcal{X})$$

The other KL

What about the 'other' KL ($q = \operatorname{argmin} \mathbf{KL}[P \| q]$)?

For a factored approximation the (clique) marginals obtained by minimising this KL are correct:

$$\begin{aligned}
\operatorname*{argmin}_{q_i} \mathbf{KL}\left[P(\mathcal{Y}|\mathcal{X}) \middle\| \prod q_j(\mathcal{Y}_j|\mathcal{X})\right] &= \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}\ P(\mathcal{Y}|\mathcal{X}) \log \prod_j q_j(\mathcal{Y}_j|\mathcal{X}) \\
&= \operatorname*{argmin}_{q_i} - \sum_j \int d\mathcal{Y}\ P(\mathcal{Y}|\mathcal{X}) \log q_j(\mathcal{Y}_j|\mathcal{X})
\end{aligned}$$

The other KL

What about the 'other' KL ($q = \operatorname{argmin} \mathbf{KL}[P \| q]$)?

For a factored approximation the (clique) marginals obtained by minimising this KL are correct:

$$\begin{aligned}
\operatorname*{argmin}_{q_i} \mathbf{KL}\left[P(\mathcal{Y}|\mathcal{X}) \middle\| \prod q_j(\mathcal{Y}_j|\mathcal{X})\right] &= \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}\; P(\mathcal{Y}|\mathcal{X}) \log \prod_j q_j(\mathcal{Y}_j|\mathcal{X}) \\
&= \operatorname*{argmin}_{q_i} - \sum_j \int d\mathcal{Y}\; P(\mathcal{Y}|\mathcal{X}) \log q_j(\mathcal{Y}_j|\mathcal{X}) \\
&= \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}_i\; P(\mathcal{Y}_i|\mathcal{X}) \log q_i(\mathcal{Y}_i|\mathcal{X})
\end{aligned}$$

The other KL

What about the 'other' KL ($q = \operatorname{argmin} \mathbf{KL}[P \| q]$)?

For a factored approximation the (clique) marginals obtained by minimising this KL are correct:

$$\begin{aligned}
\operatorname*{argmin}_{q_i} \mathbf{KL}\Big[P(\mathcal{Y}|\mathcal{X}) \Big\| \prod q_j(\mathcal{Y}_j|\mathcal{X})\Big] &= \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}\; P(\mathcal{Y}|\mathcal{X}) \log \prod_j q_j(\mathcal{Y}_j|\mathcal{X}) \\
&= \operatorname*{argmin}_{q_i} - \sum_j \int d\mathcal{Y}\; P(\mathcal{Y}|\mathcal{X}) \log q_j(\mathcal{Y}_j|\mathcal{X}) \\
&= \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}_i\; P(\mathcal{Y}_i|\mathcal{X}) \log q_i(\mathcal{Y}_i|\mathcal{X}) \\
&= P(\mathcal{Y}_i|\mathcal{X})
\end{aligned}$$

and the marginals are what we need for learning (although if factored over disjoint sets as in the variational approximation some cliques will be missing).

The other KL

What about the 'other' KL ($q = \operatorname{argmin} \mathbf{KL}[P \| q]$)?

For a factored approximation the (clique) marginals obtained by minimising this KL are correct:

$$\begin{aligned}
\operatorname*{argmin}_{q_i} \mathbf{KL}\left[P(\mathcal{Y}|\mathcal{X}) \middle\| \prod q_j(\mathcal{Y}_j|\mathcal{X})\right] &= \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}\; P(\mathcal{Y}|\mathcal{X}) \log \prod_j q_j(\mathcal{Y}_j|\mathcal{X}) \\
&= \operatorname*{argmin}_{q_i} - \sum_j \int d\mathcal{Y}\; P(\mathcal{Y}|\mathcal{X}) \log q_j(\mathcal{Y}_j|\mathcal{X}) \\
&= \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}_i\; P(\mathcal{Y}_i|\mathcal{X}) \log q_i(\mathcal{Y}_i|\mathcal{X}) \\
&= P(\mathcal{Y}_i|\mathcal{X})
\end{aligned}$$

and the marginals are what we need for learning (although if factored over disjoint sets as in the variational approximation some cliques will be missing).

Perversely, this means finding the best q for this KL is intractable!

The other KL

What about the 'other' KL ($q = \operatorname{argmin} \mathbf{KL}[P \| q]$)?

For a factored approximation the (clique) marginals obtained by minimising this KL are correct:

$$\begin{aligned}
\operatorname*{argmin}_{q_i} \mathbf{KL}\left[P(\mathcal{Y}|\mathcal{X}) \middle\| \prod q_j(\mathcal{Y}_j|\mathcal{X})\right] &= \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}\ P(\mathcal{Y}|\mathcal{X}) \log \prod_j q_j(\mathcal{Y}_j|\mathcal{X}) \\
&= \operatorname*{argmin}_{q_i} - \sum_j \int d\mathcal{Y}\ P(\mathcal{Y}|\mathcal{X}) \log q_j(\mathcal{Y}_j|\mathcal{X}) \\
&= \operatorname*{argmin}_{q_i} - \int d\mathcal{Y}_i\ P(\mathcal{Y}_i|\mathcal{X}) \log q_i(\mathcal{Y}_i|\mathcal{X}) \\
&= P(\mathcal{Y}_i|\mathcal{X})
\end{aligned}$$

and the marginals are what we need for learning (although if factored over disjoint sets as in the variational approximation some cliques will be missing).

Perversely, this means finding the best q for this KL is intractable!

But it raises the hope that approximate minimisation might still yield useful results.

Approximate optimisation

The posterior distribution in a graphical model is a (normalised) product of factors:

$$P(\mathcal{Y}|\mathcal{X}) = \frac{P(\mathcal{Y},\mathcal{X})}{P(\mathcal{X})} = \frac{1}{Z}\prod_i P(Y_i|\,\mathrm{pa}(Y_i)) \propto \prod_{i=1}^{N} f_i(\mathcal{Y}_i)$$

where the $\mathcal{Y}_i$ are not necessarily disjoint. In the language of EP the f_i are called sites.

Approximate optimisation

The posterior distribution in a graphical model is a (normalised) product of factors:

$$P(\mathcal{Y}|\mathcal{X}) = \frac{P(\mathcal{Y},\mathcal{X})}{P(\mathcal{X})} = \frac{1}{Z}\prod_i P(Y_i|\,\mathrm{pa}(Y_i)) \propto \prod_{i=1}^{N} f_i(\mathcal{Y}_i)$$

where the $\mathcal{Y}_i$ are not necessarily disjoint. In the language of EP the f_i are called sites.

Consider q with the same factorisation, but potentially approximated sites: $q(\mathcal{Y}) \stackrel{\text{def}}{=} \prod_{i=1}^{N} \tilde{f}_i(\mathcal{Y}_i)$.

We would like to minimise (at least in some sense) $\mathbf{KL}[P\|q]$.

Approximate optimisation

The posterior distribution in a graphical model is a (normalised) product of factors:

$$P(\mathcal{Y}|\mathcal{X}) = \frac{P(\mathcal{Y},\mathcal{X})}{P(\mathcal{X})} = \frac{1}{Z}\prod_i P(Y_i|\,\mathrm{pa}(Y_i)) \propto \prod_{i=1}^{N} f_i(\mathcal{Y}_i)$$

where the $\mathcal{Y}_i$ are not necessarily disjoint. In the language of EP the f_i are called sites.

Consider q with the same factorisation, but potentially approximated sites: $q(\mathcal{Y}) \stackrel{\text{def}}{=} \prod_{i=1}^{N} \tilde{f}_i(\mathcal{Y}_i)$.

We would like to minimise (at least in some sense) $\mathbf{KL}[P\|q]$.

Possible optimisations:

Approximate optimisation

The posterior distribution in a graphical model is a (normalised) product of factors:

$$P(\mathcal{Y}|\mathcal{X}) = \frac{P(\mathcal{Y},\mathcal{X})}{P(\mathcal{X})} = \frac{1}{Z}\prod_i P(Y_i|\,\mathrm{pa}(Y_i)) \propto \prod_{i=1}^{N} f_i(\mathcal{Y}_i)$$

where the $\mathcal{Y}_i$ are not necessarily disjoint. In the language of EP the f_i are called sites.

Consider q with the same factorisation, but potentially approximated sites: $q(\mathcal{Y}) \stackrel{\text{def}}{=} \prod_{i=1}^{N} \tilde{f}_i(\mathcal{Y}_i)$.

We would like to minimise (at least in some sense) $\mathbf{KL}[P\|q]$.

Possible optimisations:

$$\min_{q(\mathcal{Y})} \mathbf{KL}\Big[\prod_{i=1}^{N} f_i(\mathcal{Y}_i)\Big\|\prod_{i=1}^{N} \tilde{f}_i(\mathcal{Y}_i)\Big] \qquad \text{(global: intractable)}$$

Approximate optimisation

The posterior distribution in a graphical model is a (normalised) product of factors:

$$P(\mathcal{Y}|\mathcal{X}) = \frac{P(\mathcal{Y},\mathcal{X})}{P(\mathcal{X})} = \frac{1}{Z}\prod_i P(Y_i|\,\mathrm{pa}(Y_i)) \propto \prod_{i=1}^{N} f_i(\mathcal{Y}_i)$$

where the $\mathcal{Y}_i$ are not necessarily disjoint. In the language of EP the f_i are called sites.

Consider q with the same factorisation, but potentially approximated sites: $q(\mathcal{Y}) \stackrel{\mathrm{def}}{=} \prod_{i=1}^{N} \tilde{f}_i(\mathcal{Y}_i)$.

We would like to minimise (at least in some sense) $\mathbf{KL}[P\|q]$.

Possible optimisations:

$$\min_{q(\mathcal{Y})} \mathbf{KL}\left[\prod_{i=1}^{N} f_i(\mathcal{Y}_i)\middle\|\prod_{i=1}^{N} \tilde{f}_i(\mathcal{Y}_i)\right]$$ (global: intractable)

$$\min_{\tilde{f}_i} \mathbf{KL}\left[f_i(\mathcal{Y}_i)\middle\|\tilde{f}_i(\mathcal{Y}_i)\right]$$ (local, fixed: simple, inaccurate)

Approximate optimisation

The posterior distribution in a graphical model is a (normalised) product of factors:

$$P(\mathcal{Y}|\mathcal{X}) = \frac{P(\mathcal{Y},\mathcal{X})}{P(\mathcal{X})} = \frac{1}{Z}\prod_i P(Y_i|\,\mathrm{pa}(Y_i)) \propto \prod_{i=1}^{N} f_i(\mathcal{Y}_i)$$

where the $\mathcal{Y}_i$ are not necessarily disjoint. In the language of EP the f_i are called sites.

Consider q with the same factorisation, but potentially approximated sites: $q(\mathcal{Y}) \stackrel{\text{def}}{=} \prod_{i=1}^{N} \tilde{f}_i(\mathcal{Y}_i)$.

We would like to minimise (at least in some sense) $\mathbf{KL}[P\|q]$.

Possible optimisations:

$$\min_{q(\mathcal{Y})} \mathbf{KL}\left[\prod_{i=1}^{N} f_i(\mathcal{Y}_i)\middle\|\prod_{i=1}^{N} \tilde{f}_i(\mathcal{Y}_i)\right]$$ (global: intractable)

$$\min_{\tilde{f}_i} \mathbf{KL}\left[f_i(\mathcal{Y}_i)\middle\|\tilde{f}_i(\mathcal{Y}_i)\right]$$ (local, fixed: simple, inaccurate)

$$\min_{\tilde{f}_i} \mathbf{KL}\left[f_i(\mathcal{Y}_i)\prod_{j\neq i}\tilde{f}_j(\mathcal{Y}_j)\middle\|\tilde{f}_i(\mathcal{Y}_i)\prod_{j\neq i}\tilde{f}_j(\mathcal{Y}_j)\right]$$ (local, contextual: iterative, accurate)

Approximate optimisation

The posterior distribution in a graphical model is a (normalised) product of factors:

$$P(\mathcal{Y}|\mathcal{X}) = \frac{P(\mathcal{Y},\mathcal{X})}{P(\mathcal{X})} = \frac{1}{Z}\prod_i P(Y_i|\,\mathrm{pa}(Y_i)) \propto \prod_{i=1}^{N} f_i(\mathcal{Y}_i)$$

where the $\mathcal{Y}_i$ are not necessarily disjoint. In the language of EP the f_i are called sites.

Consider q with the same factorisation, but potentially approximated sites: $q(\mathcal{Y}) \stackrel{\text{def}}{=} \prod_{i=1}^{N} \tilde{f}_i(\mathcal{Y}_i)$.

We would like to minimise (at least in some sense) $\mathbf{KL}[P\|q]$.

Possible optimisations:

$$\min_{q(\mathcal{Y})} \mathbf{KL}\left[\prod_{i=1}^{N} f_i(\mathcal{Y}_i)\middle\|\prod_{i=1}^{N}\tilde{f}_i(\mathcal{Y}_i)\right]$$ (global: intractable)

$$\min_{\tilde{f}_i} \mathbf{KL}\left[f_i(\mathcal{Y}_i)\middle\|\tilde{f}_i(\mathcal{Y}_i)\right]$$ (local, fixed: simple, inaccurate)

$$\min_{\tilde{f}_i} \mathbf{KL}\left[f_i(\mathcal{Y}_i)\prod_{j\neq i}\tilde{f}_j(\mathcal{Y}_j)\middle\|\tilde{f}_i(\mathcal{Y}_i)\prod_{j\neq i}\tilde{f}_j(\mathcal{Y}_j)\right]$$ (local, contextual: iterative, accurate) $\leftarrow$ EP

Expectation? Propagation?

EP is really two ideas:

- Approximation of factors.

Expectation? Propagation?

EP is really two ideas:

- Approximation of factors.
  - Usually by “projection” to exponential families.
  - This involves finding expected sufficient statistics, hence expectation.

Expectation? Propagation?

EP is really two ideas:

- Approximation of factors.
  - Usually by “projection” to exponential families.
  - This involves finding expected sufficient statistics, hence expectation.
- Local divergence minimization in the context of other factors.

Expectation? Propagation?

EP is really two ideas:

- Approximation of factors.
  - Usually by “projection” to exponential families.
  - This involves finding expected sufficient statistics, hence expectation.
- Local divergence minimization in the context of other factors.
  - This leads to a message passing approach, hence propagation.

Local updates

Each EP update involves a KL minimisation:

$$\tilde{f}_i^{\text{new}}(\mathcal{Y}) \leftarrow \underset{f \in \{\tilde{f}\}}{\operatorname{argmin}}\ \mathbf{KL}[f_i(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}) \| f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y})]$$

Write $q_{\neg i}(\mathcal{Y}) = q_{\neg i}(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}_{\neg i} | \mathcal{Y}_i)$. Then:

$$\begin{aligned}
&\min_f \mathbf{KL}[f_i(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}) \| f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y})] \\
&\qquad = \max_f \int d\mathcal{Y}_i d\mathcal{Y}_{\neg i}\ f_i(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}) \log f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}) \\
&\qquad = \max_f \int d\mathcal{Y}_i d\mathcal{Y}_{\neg i}\ f_i(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}_{\neg i}|\mathcal{Y}_i) \big(\log f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}_i) + \log q_{\neg i}(\mathcal{Y}_{\neg i}|\mathcal{Y}_i) \big) \\
&\qquad = \max_f \int d\mathcal{Y}_i\ f_i(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}_i) \big(\log f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}_i) \big) \int d\mathcal{Y}_{\neg i}\ q_{\neg i}(\mathcal{Y}_{\neg i}|\mathcal{Y}_i) \\
&\qquad = \min_f \mathbf{KL}[f_i(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}_i) \| f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}_i)]
\end{aligned}$$

$q_{\neg i}(\mathcal{Y}_i)$ is sometimes called the cavity distribution.

Expectation Propagation (EP)

Input $f_1(\mathcal{Y}_1) \dots f_N(\mathcal{Y}_N)$

Initialize $\tilde{f}_1(\mathcal{Y}_1) = \underset{f \in \{\tilde{f}\}}{\operatorname{argmin}} \, \mathbf{KL}[f_1(\mathcal{Y}_1) \| f_1(\mathcal{Y}_1)]$, $\tilde{f}_i(\mathcal{Y}_i) = 1$ for $i > 1$, $q(\mathcal{Y}) \propto \prod_i \tilde{f}_i(\mathcal{Y}_i)$

repeat

for $i = 1 \dots N$ **do**

Delete: $q_{\neg i}(\mathcal{Y}) \leftarrow \dfrac{q(\mathcal{Y})}{\tilde{f}_i(\mathcal{Y}_i)} = \prod_{j \neq i} \tilde{f}_j(\mathcal{Y}_j)$

Project: $\tilde{f}_i^{\mathrm{new}}(\mathcal{Y}) \leftarrow \underset{f \in \{\tilde{f}\}}{\operatorname{argmin}} \, \mathbf{KL}[f_i(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}) \| f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y})]$

Include: $q(\mathcal{Y}) \leftarrow \tilde{f}_i^{\mathrm{new}}(\mathcal{Y}_i) \, q_{\neg i}(\mathcal{Y})$

end for

until convergence

Message Passing

- The cavity distribution (in a tree) can be further broken down into a product of terms from each neighbouring clique:

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \mathrm{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

Message Passing

- The cavity distribution (in a tree) can be further broken down into a product of terms from each neighbouring clique:

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \mathrm{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

- Once the *i*th site has been approximated, the messages can be passed on to neighbouring cliques by marginalising to the shared variables (SSM example follows). $\Rightarrow$ belief propagation.

Message Passing

- The cavity distribution (in a tree) can be further broken down into a product of terms from each neighbouring clique:

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \text{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

- Once the ith site has been approximated, the messages can be passed on to neighbouring cliques by marginalising to the shared variables (SSM example follows). $\Rightarrow$ belief propagation.

- In loopy graphs, we can use loopy belief propagation. In that case

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \text{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

  becomes an approximation to the **true** cavity distribution (or we can recast the approximation directly in terms of messages $\Rightarrow$ later lecture).

Message Passing

- The cavity distribution (in a tree) can be further broken down into a product of terms from each neighbouring clique:

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \mathrm{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

- Once the ith site has been approximated, the messages can be passed on to neighbouring cliques by marginalising to the shared variables (SSM example follows). $\Rightarrow$ belief propagation.

- In loopy graphs, we can use loopy belief propagation. In that case

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \mathrm{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

  becomes an approximation to the **true** cavity distribution (or we can recast the approximation directly in terms of messages $\Rightarrow$ later lecture).
- For some approximations (e.g. Gaussian) may be able to compute true loopy cavity using approximate sites, even if computing exact message would have been intractable.

Message Passing

- The cavity distribution (in a tree) can be further broken down into a product of terms from each neighbouring clique:

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \mathrm{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

- Once the ith site has been approximated, the messages can be passed on to neighbouring cliques by marginalising to the shared variables (SSM example follows). $\Rightarrow$ belief propagation.

- In loopy graphs, we can use loopy belief propagation. In that case

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \mathrm{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

  becomes an approximation to the **true** cavity distribution (or we can recast the approximation directly in terms of messages $\Rightarrow$ later lecture).
- For some approximations (e.g. Gaussian) may be able to compute true loopy cavity using approximate sites, even if computing exact message would have been intractable.
- In either case, message updates can be scheduled in any order.

Message Passing

- The cavity distribution (in a tree) can be further broken down into a product of terms from each neighbouring clique:

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \mathrm{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

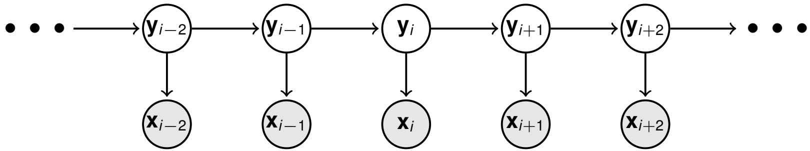
- Once the ith site has been approximated, the messages can be passed on to neighbouring cliques by marginalising to the shared variables (SSM example follows). $\Rightarrow$ belief propagation.

- In loopy graphs, we can use loopy belief propagation. In that case

$$q_{\neg i}(\mathcal{Y}_i) = \prod_{j \in \mathrm{ne}(i)} M_{j \to i}(\mathcal{Y}_j \cap \mathcal{Y}_i)$$

  becomes an approximation to the **true** cavity distribution (or we can recast the approximation directly in terms of messages $\Rightarrow$ later lecture).
- For some approximations (e.g. Gaussian) may be able to compute true loopy cavity using approximate sites, even if computing exact message would have been intractable.
- In either case, message updates can be scheduled in any order.
- No guarantee of convergence (but see “power-EP” methods).

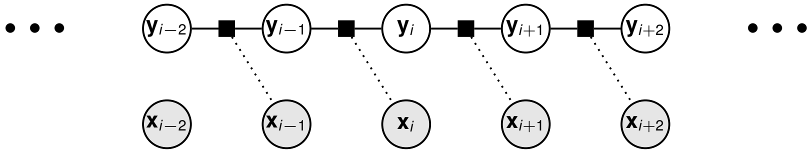
EP for a NLSSM



$$P(\mathbf{y}_i|\mathbf{y}_{i-1}) = \phi_i(\mathbf{y}_i, \mathbf{y}_{i-1}) \qquad \textit{e.g.}\ \exp(-\|\mathbf{y}_i - h_s(\mathbf{y}_{i-1})\|^2/2\sigma^2)$$

$$P(\mathbf{x}_i|\mathbf{y}_i) = \psi_i(\mathbf{y}_i) \qquad \textit{e.g.}\ \exp(-\|\mathbf{x}_i - h_o(\mathbf{y}_i)\|^2/2\sigma^2)$$

EP for a NLSSM

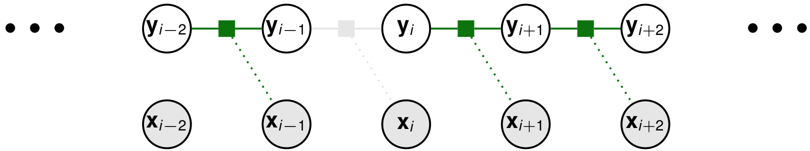


$$P(\mathbf{y}_i|\mathbf{y}_{i-1}) = \phi_i(\mathbf{y}_i, \mathbf{y}_{i-1}) \qquad \textit{e.g.}\ \exp(-\|\mathbf{y}_i - h_s(\mathbf{y}_{i-1})\|^2/2\sigma^2)$$

$$P(\mathbf{x}_i|\mathbf{y}_i) = \psi_i(\mathbf{y}_i) \qquad \textit{e.g.}\ \exp(-\|\mathbf{x}_i - h_o(\mathbf{y}_i)\|^2/2\sigma^2)$$

Then $f_i(\mathbf{y}_i, \mathbf{y}_{i-1}) = \phi_i(\mathbf{y}_i, \mathbf{y}_{i-1})\psi_i(\mathbf{y}_i)$. As ϕ_i and ψ_i are non-linear, inference is not generally tractable.

EP for a NLSSM



$$P(\mathbf{y}_i|\mathbf{y}_{i-1}) = \phi_i(\mathbf{y}_i, \mathbf{y}_{i-1}) \qquad \textit{e.g. } \exp(-\|\mathbf{y}_i - h_s(\mathbf{y}_{i-1})\|^2/2\sigma^2)$$

$$P(\mathbf{x}_i|\mathbf{y}_i) = \psi_i(\mathbf{y}_i) \qquad \textit{e.g. } \exp(-\|\mathbf{x}_i - h_o(\mathbf{y}_i)\|^2/2\sigma^2)$$

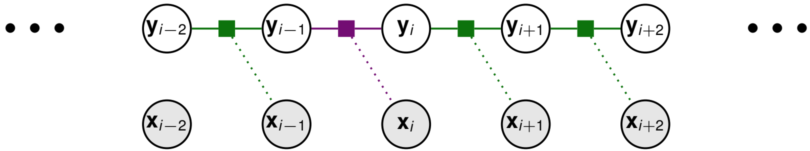
Then $f_i(\mathbf{y}_i, \mathbf{y}_{i-1}) = \phi_i(\mathbf{y}_i, \mathbf{y}_{i-1})\psi_i(\mathbf{y}_i)$. As ϕ_i and ψ_i are non-linear, inference is not generally tractable.

Assume $\tilde{f}_i(\mathbf{y}_i, \mathbf{y}_{i-1})$ is Gaussian. Then,

$$q_{\neg i}(\mathbf{y}_i, \mathbf{y}_{i-1}) = \sum_{\substack{\mathbf{y}_1 \ldots \mathbf{y}_{i-2} \\ \mathbf{y}_{i+1} \ldots \mathbf{y}_l}} \prod_{i' \neq i} \tilde{f}_{i'}(\mathbf{y}_{i'}, \mathbf{y}_{i'-1}) = \underbrace{\sum_{\mathbf{y}_1 \ldots \mathbf{y}_{i-2}} \prod_{i' < i} \tilde{f}_{i'}(\mathbf{y}_{i'}, \mathbf{y}_{i'-1})}_{\alpha_{i-1}(\mathbf{y}_{i-1})} \underbrace{\sum_{\mathbf{y}_{i+1} \ldots \mathbf{y}_l} \prod_{i' > i} \tilde{f}_{i'}(\mathbf{y}_{i'}, \mathbf{y}_{i'-1})}_{\beta_i(\mathbf{y}_i)}$$

with both α and β Gaussian.

EP for a NLSSM



$$P(\mathbf{y}_i|\mathbf{y}_{i-1}) = \phi_i(\mathbf{y}_i, \mathbf{y}_{i-1}) \qquad \textit{e.g. } \exp(-\|\mathbf{y}_i - h_s(\mathbf{y}_{i-1})\|^2/2\sigma^2)$$

$$P(\mathbf{x}_i|\mathbf{y}_i) = \psi_i(\mathbf{y}_i) \qquad \textit{e.g. } \exp(-\|\mathbf{x}_i - h_o(\mathbf{y}_i)\|^2/2\sigma^2)$$

Then $f_i(\mathbf{y}_i, \mathbf{y}_{i-1}) = \phi_i(\mathbf{y}_i, \mathbf{y}_{i-1})\psi_i(\mathbf{y}_i)$. As ϕ_i and ψ_i are non-linear, inference is not generally tractable.
Assume $\tilde{f}_i(\mathbf{y}_i, \mathbf{y}_{i-1})$ is Gaussian. Then,

$$q_{\neg i}(\mathbf{y}_i, \mathbf{y}_{i-1}) = \sum_{\substack{\mathbf{y}_1 \ldots \mathbf{y}_{i-2} \\ \mathbf{y}_{i+1} \ldots \mathbf{y}_I}} \prod_{i' \neq i} \tilde{f}_{i'}(\mathbf{y}_{i'}, \mathbf{y}_{i'-1}) = \underbrace{\sum_{\mathbf{y}_1 \ldots \mathbf{y}_{i-2}} \prod_{i' < i} \tilde{f}_{i'}(\mathbf{y}_{i'}, \mathbf{y}_{i'-1})}_{\alpha_{i-1}(\mathbf{y}_{i-1})} \underbrace{\sum_{\mathbf{y}_{i+1} \ldots \mathbf{y}_I} \prod_{i' > i} \tilde{f}_{i'}(\mathbf{y}_{i'}, \mathbf{y}_{i'-1})}_{\beta_i(\mathbf{y}_i)}$$

with both α and β Gaussian.

$$\tilde{f}_i(\mathbf{y}_i, \mathbf{y}_{i-1}) = \operatorname*{argmin}_{f \in \mathcal{N}} \mathbf{KL}\Big[\phi_i(\mathbf{y}_i, \mathbf{y}_{i-1})\psi_i(\mathbf{y}_i)\alpha_{i-1}(\mathbf{y}_{i-1})\beta_i(\mathbf{y}_i) \Big\| f(\mathbf{y}_i, \mathbf{y}_{i-1})\alpha_{i-1}(\mathbf{y}_{i-1})\beta_i(\mathbf{y}_i)\Big]$$

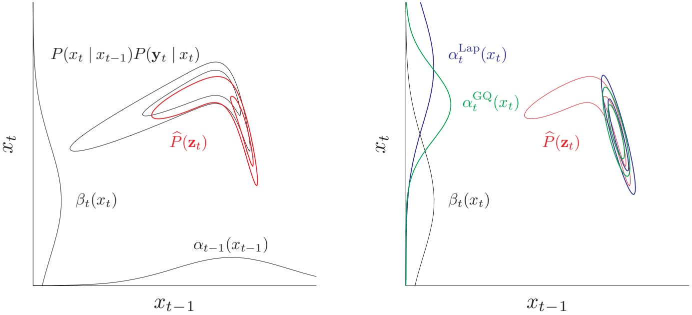
NLSSM EP message updates

$$\tilde{f}_i(\mathbf{y}_i, \mathbf{y}_{i-1}) = \underset{f \in \mathcal{N}}{\operatorname{argmin}} \, \mathbf{KL}\Big[\underbrace{\phi_i(\mathbf{y}_i, \mathbf{y}_{i-1})\psi_i(\mathbf{y}_i)\alpha_{i-1}(\mathbf{y}_{i-1})\beta_i(\mathbf{y}_i)}_{\widehat{P}(\mathbf{y}_{i-1}, \mathbf{y}_i)} \Big\| \underbrace{f(\mathbf{y}_i, \mathbf{y}_{i-1})\alpha_{i-1}(\mathbf{y}_{i-1})\beta_i(\mathbf{y}_i)}_{\tilde{P}(\mathbf{y}_{i-1}, \mathbf{y}_i)}\Big]$$

$$\alpha_i(\mathbf{y}_i) = \tilde{P}(\mathbf{y}_{i-1}, \mathbf{y}_i)/\beta_i(\mathbf{y}_i)$$

$$\beta_{i-1}(\mathbf{y}_{i-1}) = \tilde{P}(\mathbf{y}_{i-1}, \mathbf{y}_i)/\alpha_{i-1}(\mathbf{y}_{i-1})$$

For exponential family approximations, division $\Rightarrow$ subtraction of natural parameters.



[Note: For this figure (from a paper): x is latent, $\mathbf{y}$ observed and $\mathbf{z}_t = [x_{t-1}, x_t]$.]

Moment Matching

Each EP update involves a KL minimisation:

$$\tilde{f}_i^{\text{new}}(\mathcal{Y}) \leftarrow \underset{f \in \{\tilde{f}\}}{\operatorname{argmin}} \, \mathbf{KL}[f_i(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y}) \| f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y})]$$

Usually, both $q_{\neg i}(\mathcal{Y}_i)$ and $\tilde{f}$ are in the same exponential family. Let $q(x) = \frac{1}{Z(\boldsymbol{\theta})} e^{\mathsf{S}(x) \cdot \boldsymbol{\theta}}$. Then

$$\begin{aligned}
\underset{q}{\operatorname{argmin}} \, \mathbf{KL}\big[p(x) \big\| q(x)\big] &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \, \mathbf{KL}\left[p(x) \middle\| \frac{1}{Z(\boldsymbol{\theta})} e^{\mathsf{T}(x) \cdot \boldsymbol{\theta}}\right] \\
&= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} - \int dx \; p(x) \log \frac{1}{Z(\boldsymbol{\theta})} e^{\mathsf{T}(x) \cdot \boldsymbol{\theta}} \\
&= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} - \int dx \; p(x) \mathsf{T}(x) \cdot \boldsymbol{\theta} + \log Z(\boldsymbol{\theta}) \\
\frac{\partial}{\partial \boldsymbol{\theta}} &= - \int dx \; p(x) \mathsf{T}(x) + \frac{1}{Z(\boldsymbol{\theta})} \frac{\partial}{\partial \boldsymbol{\theta}} \int dx \; e^{\mathsf{T}(x) \cdot \boldsymbol{\theta}} \\
&= - \langle \mathsf{T}(x) \rangle_p + \frac{1}{Z(\boldsymbol{\theta})} \int dx \; e^{\mathsf{T}(x) \cdot \boldsymbol{\theta}} \mathsf{T}(x) \\
&= - \langle \mathsf{T}(x) \rangle_p + \langle \mathsf{T}(x) \rangle_q
\end{aligned}$$

So minimum is found by matching sufficient stats. This is usually moment matching.

Numerical issues

How do we calculate $\langle \mathrm{T}(x) \rangle_p$?

Numerical issues

How do we calculate $\langle \mathrm{T}(x) \rangle_p$?

Often analytically tractable, but even if not requires a (relatively) low-dimensional integral:

- Quadrature methods.

Numerical issues

How do we calculate $\langle \mathrm{T}(x) \rangle_p$?

Often analytically tractable, but even if not requires a (relatively) low-dimensional integral:

- Quadrature methods.
  - Classical Gaussian quadrature (same Gauss, but nothing to do with the distribution) gives an iterative version of Sigma-point methods.

Numerical issues

How do we calculate $\langle \mathrm{T}(x) \rangle_p$?

Often analytically tractable, but even if not requires a (relatively) low-dimensional integral:

- Quadrature methods.
  - Classical Gaussian quadrature (same Gauss, but nothing to do with the distribution) gives an iterative version of Sigma-point methods.
  - Positive definite joints, but not guaranteed to give positive definite messages.

Numerical issues

How do we calculate $\langle \mathrm{T}(x) \rangle_p$?

Often analytically tractable, but even if not requires a (relatively) low-dimensional integral:

- Quadrature methods.
  - Classical Gaussian quadrature (same Gauss, but nothing to do with the distribution) gives an iterative version of Sigma-point methods.
  - Positive definite joints, but not guaranteed to give positive definite messages.
  - Heuristics include skipping non-positive-definite steps, or damping messages by interpolation or exponentiating to power < 1.

Numerical issues

How do we calculate $\langle \mathrm{T}(x) \rangle_p$?

Often analytically tractable, but even if not requires a (relatively) low-dimensional integral:

- Quadrature methods.
  - Classical Gaussian quadrature (same Gauss, but nothing to do with the distribution) gives an iterative version of Sigma-point methods.
  - Positive definite joints, but not guaranteed to give positive definite messages.
  - Heuristics include skipping non-positive-definite steps, or damping messages by interpolation or exponentiating to power < 1.
  - Other quadrature approaches (*e.g.* GP quadrature) may be more accurate, and may allow formal consttaint to PD cone.

Numerical issues

How do we calculate $\langle \mathrm{T}(x) \rangle_p$?

Often analytically tractable, but even if not requires a (relatively) low-dimensional integral:

- Quadrature methods.
  - Classical Gaussian quadrature (same Gauss, but nothing to do with the distribution) gives an iterative version of Sigma-point methods.
  - Positive definite joints, but not guaranteed to give positive definite messages.
  - Heuristics include skipping non-positive-definite steps, or damping messages by interpolation or exponentiating to power < 1.
  - Other quadrature approaches (*e.g.* GP quadrature) may be more accurate, and may allow formal consttaint to PD cone.

- Laplace approximation.

Numerical issues

How do we calculate $\langle \mathrm{T}(x) \rangle_p$?

Often analytically tractable, but even if not requires a (relatively) low-dimensional integral:

- Quadrature methods.
  - Classical Gaussian quadrature (same Gauss, but nothing to do with the distribution) gives an iterative version of Sigma-point methods.
  - Positive definite joints, but not guaranteed to give positive definite messages.
  - Heuristics include skipping non-positive-definite steps, or damping messages by interpolation or exponentiating to power < 1.
  - Other quadrature approaches (*e.g.* GP quadrature) may be more accurate, and may allow formal consttaint to PD cone.
- Laplace approximation.
  - Equivalent to Laplace propagation.

Numerical issues

How do we calculate $\langle \mathrm{T}(x) \rangle_p$?

Often analytically tractable, but even if not requires a (relatively) low-dimensional integral:

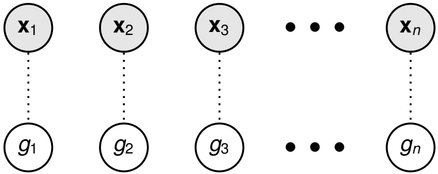
- Quadrature methods.
  - Classical Gaussian quadrature (same Gauss, but nothing to do with the distribution) gives an iterative version of Sigma-point methods.
  - Positive definite joints, but not guaranteed to give positive definite messages.
  - Heuristics include skipping non-positive-definite steps, or damping messages by interpolation or exponentiating to power < 1.
  - Other quadrature approaches (*e.g.* GP quadrature) may be more accurate, and may allow formal consttaint to PD cone.
- Laplace approximation.
  - Equivalent to Laplace propagation.
  - As long as messages remain positive definite will converge to global Laplace approximation.

EP for Gaussian process classification

EP provides the most succesful framework for Gaussian-process modelling of non-Gaussian observations (*e.g.* for classification).

EP for Gaussian process classification

EP provides the most succesful framework for Gaussian-process modelling of non-Gaussian observations (*e.g.* for classification).

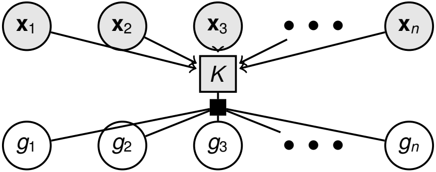


Recall:

- A GP defines a multivariate Gaussian distribution on any finite subset of random vars $\{g_1 \dots g_n\}$ drawn from a (usually uncountable) potential set indexed by "inputs" $\mathbf{x}_i$.

EP for Gaussian process classification

EP provides the most succesful framework for Gaussian-process modelling of non-Gaussian observations (*e.g.* for classification).

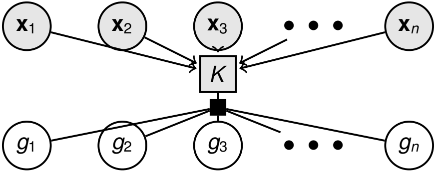


Recall:

- A GP defines a multivariate Gaussian distribution on any finite subset of random vars $\{g_1 \dots g_n\}$ drawn from a (usually uncountable) potential set indexed by "inputs" $\mathbf{x}_i$.
- The Gaussian parameters depend on the inputs: ($\boldsymbol{\mu} = [\mu(\mathbf{x}_i)],\ \Sigma = [K(\mathbf{x}_i, \mathbf{x}_j)]$).

EP for Gaussian process classification

EP provides the most succesful framework for Gaussian-process modelling of non-Gaussian observations (*e.g.* for classification).

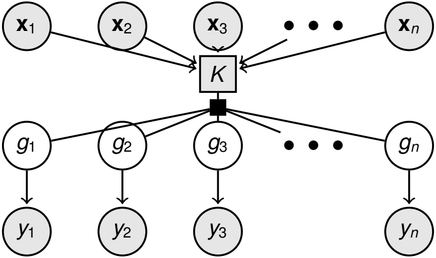


Recall:

- A GP defines a multivariate Gaussian distribution on any finite subset of random vars $\{g_1 \dots g_n\}$ drawn from a (usually uncountable) potential set indexed by "inputs" $\mathbf{x}_i$.
- The Gaussian parameters depend on the inputs: $(\boldsymbol{\mu} = [\mu(\mathbf{x}_i)],\ \Sigma = [K(\mathbf{x}_i, \mathbf{x}_j)])$.
- If we think of the gs as function values, a GP provides a prior over functions.

EP for Gaussian process classification

EP provides the most succesful framework for Gaussian-process modelling of non-Gaussian observations (*e.g.* for classification).

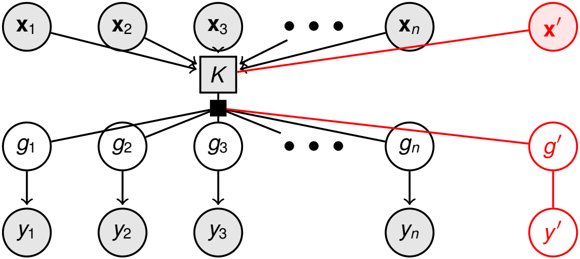


Recall:

- A GP defines a multivariate Gaussian distribution on any finite subset of random vars $\{g_1 \ldots g_n\}$ drawn from a (usually uncountable) potential set indexed by "inputs" $\mathbf{x}_i$.
- The Gaussian parameters depend on the inputs: $(\boldsymbol{\mu} = [\mu(\mathbf{x}_i)],\ \Sigma = [K(\mathbf{x}_i, \mathbf{x}_j)])$.
- If we think of the gs as function values, a GP provides a prior over functions.
- In a GP regression model, noisy observations y_i are conditionally independent given g_i.

EP for Gaussian process classification

EP provides the most successful framework for Gaussian-process modelling of non-Gaussian observations (*e.g.* for classification).

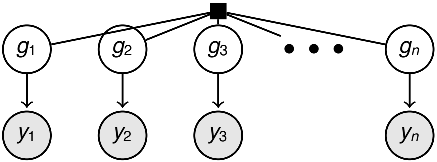


Recall:

- A GP defines a multivariate Gaussian distribution on any finite subset of random vars $\{g_1 \dots g_n\}$ drawn from a (usually uncountable) potential set indexed by "inputs" $\mathbf{x}_i$.
- The Gaussian parameters depend on the inputs: $(\boldsymbol{\mu} = [\mu(\mathbf{x}_i)],\ \Sigma = [K(\mathbf{x}_i, \mathbf{x}_j)])$.
- If we think of the gs as function values, a GP provides a prior over functions.
- In a GP regression model, noisy observations y_i are conditionally independent given g_i.
- No parameters to learn (though often hyperparameters); instead, we make predictions on test data directly: [assuming $\mu = 0$, and matrix Σ incorporates diagonal noise]

$$P(y'|\mathbf{x}', \mathcal{D}) = \mathcal{N}\left(\Sigma_{x',X}\Sigma_{X,X}^{-1}\mathbf{y},\ \Sigma_{x',x'} - \Sigma_{x',X}\Sigma_{X,X}^{-1}\Sigma_{X,x'}\right)$$

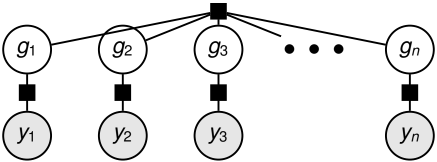
GP EP updates



- We can write the GP joint on g_i and y_i as a factor graph:

$$P(g_1 \dots g_n, y_1, \dots y_n) = \mathcal{N}(g_1 \dots g_n | \mathbf{0}, K) \prod_i \mathcal{N}(y_i | g_i, \sigma_i^2)$$

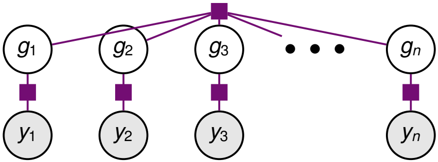
GP EP updates



- We can write the GP joint on g_i and y_i as a factor graph:

$$P(g_1 \dots g_n, y_1, \dots y_n) = \underbrace{\mathcal{N}\left(g_1 \dots g_n | \mathbf{0}, K\right)}_{f_0(\mathcal{G})} \prod_i \underbrace{\mathcal{N}\left(y_i | g_i, \sigma_i^2\right)}_{f_i(g_i)}$$

GP EP updates

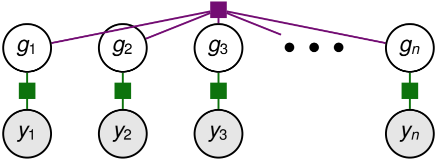


- We can write the GP joint on g_i and y_i as a factor graph:

$$P(g_1 \dots g_n, y_1, \dots y_n) = \underbrace{\mathcal{N}(g_1 \dots g_n | \mathbf{0}, K)}_{f_0(\mathcal{G})} \prod_i \underbrace{\mathcal{N}(y_i | g_i, \sigma_i^2)}_{f_i(g_i)}$$

- The same factorisation applies to non-Gaussian $P(y_i|g_i)$ (*e.g.* $P(y_i{=}1) = 1/(1 + e^{-g_i})$).

GP EP updates

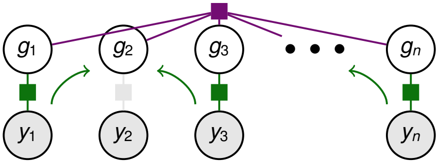


- We can write the GP joint on g_i and y_i as a factor graph:

$$P(g_1 \ldots g_n, y_1, \ldots y_n) = \underbrace{\mathcal{N}(g_1 \ldots g_n | \mathbf{0}, K)}_{f_0(\mathcal{G})} \prod_i \underbrace{\mathcal{N}(y_i | g_i, \sigma_i^2)}_{f_i(g_i)}$$

- The same factorisation applies to non-Gaussian $P(y_i|g_i)$ (*e.g.* $P(y_i{=}1) = 1/(1 + e^{-g_i})$).
- EP: approximate non-Gaussian $f_i(g_i)$ by Gaussian $\tilde{f}_i(g_i) = \mathcal{N}\left(\tilde{\mu}_i, \tilde{\psi}_i^2\right)$.

GP EP updates



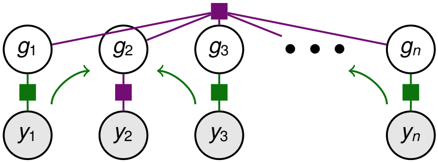
- We can write the GP joint on g_i and y_i as a factor graph:

$$P(g_1 \ldots g_n, y_1, \ldots y_n) = \underbrace{\mathcal{N}(g_1 \ldots g_n | \mathbf{0}, K)}_{f_0(\mathcal{G})} \prod_i \underbrace{\mathcal{N}(y_i | g_i, \sigma_i^2)}_{f_i(g_i)}$$

- The same factorisation applies to non-Gaussian $P(y_i|g_i)$ (*e.g.* $P(y_i{=}1) = 1/(1 + e^{-g_i})$).
- EP: approximate non-Gaussian $f_i(g_i)$ by Gaussian $\tilde{f}_i(g_i) = \mathcal{N}\left(\tilde{\mu}_i, \tilde{\psi}_i^2\right)$.
- $q_{\neg i}(g_i)$ can be constructed by the usual GP marginalisation. If $\Sigma = K + \text{diag}\left[\tilde{\psi}_1^2 \ldots \tilde{\psi}_n^2\right]$

$$q_{\neg i}(g_i) = \mathcal{N}\left(\Sigma_{i,\neg i}\Sigma_{\neg i,\neg i}^{-1}\tilde{\boldsymbol{\mu}}_{\neg i},\; K_{i,i} - \Sigma_{i,\neg i}\Sigma_{\neg i,\neg i}^{-1}\Sigma_{\neg i,i}\right)$$

GP EP updates



- We can write the GP joint on g_i and y_i as a factor graph:

$$P(g_1 \dots g_n, y_1, \dots y_n) = \underbrace{\mathcal{N}(g_1 \dots g_n | \mathbf{0}, K)}_{f_0(\mathcal{G})} \prod_i \underbrace{\mathcal{N}(y_i | g_i, \sigma_i^2)}_{f_i(g_i)}$$

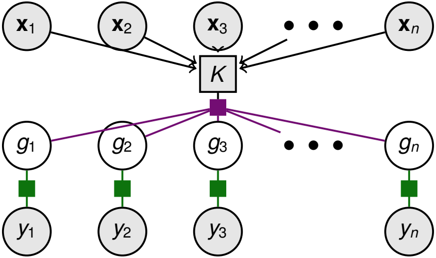
- The same factorisation applies to non-Gaussian $P(y_i|g_i)$ (*e.g.* $P(y_i{=}1) = 1/(1 + e^{-g_i})$).
- EP: approximate non-Gaussian $f_i(g_i)$ by Gaussian $\tilde{f}_i(g_i) = \mathcal{N}\left(\tilde{\mu}_i, \tilde{\psi}_i^2\right)$.
- $q_{\neg i}(g_i)$ can be constructed by the usual GP marginalisation. If $\Sigma = K + \text{diag}\left[\tilde{\psi}_1^2 \dots \tilde{\psi}_n^2\right]$

$$q_{\neg i}(g_i) = \mathcal{N}\left(\Sigma_{i,\neg i}\Sigma_{\neg i,\neg i}^{-1}\tilde{\boldsymbol{\mu}}_{\neg i},\ K_{i,i} - \Sigma_{i,\neg i}\Sigma_{\neg i,\neg i}^{-1}\Sigma_{\neg i,i}\right)$$

- The EP updates thus require calculating Gaussian expectations of $f_i(g)g^{\{1,2\}}$:

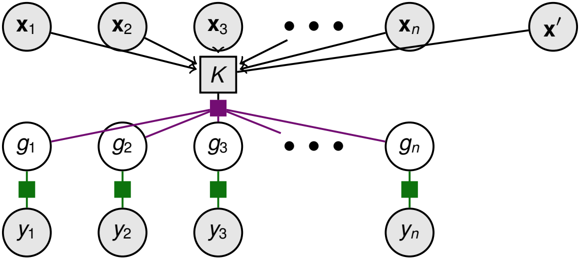
$$\tilde{f}_i^{\text{new}}(g_i) = \mathcal{N}\left(\int dg\, q_{\neg i}(g) f_i(g) g,\ \int dg\, q_{\neg i}(g) f_i(g) g^2 - (\tilde{\mu}_i^{\text{new}})^2\right)$$

EP GP prediction



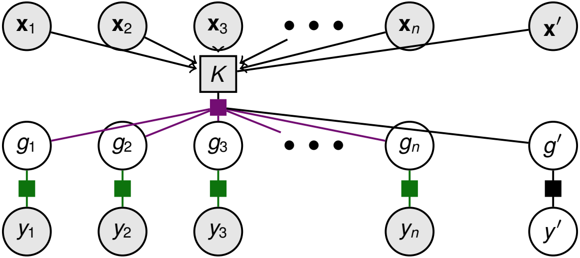
- Once appoximate site potentials have stabilised, they can be used to make predictions.

EP GP prediction



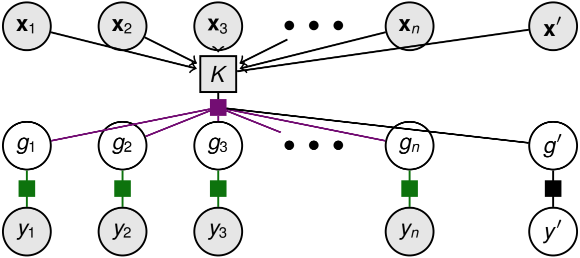
- Once appoximate site potentials have stabilised, they can be used to make predictions.
- Introducing a test point changes K, but does not affect the *marginal* $P(g_1 \ldots g_n)$ (by consistency of the GP).

EP GP prediction



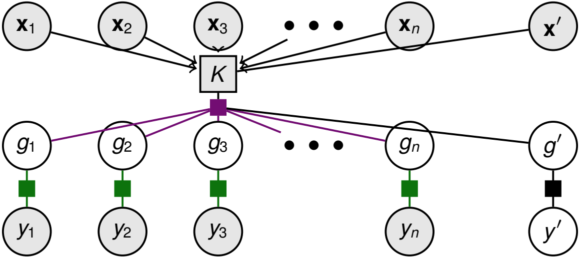
- Once appoximate site potentials have stabilised, they can be used to make predictions.
- Introducing a test point changes K, but does not affect the *marginal* $P(g_1 \dots g_n)$ (by consistency of the GP).
- The unobserved output factor provides no information about g' ($\Rightarrow$ constant factor on g')

EP GP prediction



- Once appoximate site potentials have stabilised, they can be used to make predictions.
- Introducing a test point changes K, but does not affect the *marginal* $P(g_1 \ldots g_n)$ (by consistency of the GP).
- The unobserved output factor provides no information about g' ($\Rightarrow$ constant factor on g')
- Thus no change is needed to the approximating potentials $\tilde{f}_i$.

EP GP prediction



- Once appoximate site potentials have stabilised, they can be used to make predictions.
- Introducing a test point changes K, but does not affect the *marginal* $P(g_1 \dots g_n)$ (by consistency of the GP).
- The unobserved output factor provides no information about g' ($\Rightarrow$ constant factor on g')
- Thus no change is needed to the approximating potentials $\tilde{f}_i$.
- Predictions are obtained by marginalising the approximation: [let $\tilde{\Psi} = \text{diag}[\tilde{\psi}_1^2 \dots \tilde{\psi}_n^2]$]

$$P(y'|\mathbf{x}', \mathcal{D}) = \int dg' \, P(y'|g')\mathcal{N}\Big(g' \mid K_{x',X}(K_{X,X} + \tilde{\Psi})^{-1}\tilde{\boldsymbol{\mu}}, \\ K_{x',x'} - K_{x',X}(K_{X,X} + \tilde{\Psi})^{-1}K_{X,x'}\Big)$$

Normalisers

- Approximate sites determined by moment matching are naturally normalised.

Normalisers

- Approximate sites determined by moment matching are naturally normalised.
- For posteriors, sufficient to normalise product after convergence.

Normalisers

- Approximate sites determined by moment matching are naturally normalised.
- For posteriors, sufficient to normalise product after convergence.
  - Often straightforward for exponential family approximations.

Normalisers

- Approximate sites determined by moment matching are naturally normalised.
- For posteriors, sufficient to normalise product after convergence.
  - Often straightforward for exponential family approximations.
- To compute likelihood need to keep track of site integrals:

Normalisers

- Approximate sites determined by moment matching are naturally normalised.
- For posteriors, sufficient to normalise product after convergence.
  - Often straightforward for exponential family approximations.
- To compute likelihood need to keep track of site integrals:
  - minimising "unnormalised KL":

$$\mathbf{KL}[p\|q] = \int dx\, p(x) \log \frac{p(x)}{q(x)} + \int dx\, \big(q(x) - p(x)\big)$$

incorporates normaliser into each $\tilde{f}$ (match zeroth moment, along with suff stats).

Normalisers

- Approximate sites determined by moment matching are naturally normalised.
- For posteriors, sufficient to normalise product after convergence.
  - Often straightforward for exponential family approximations.
- To compute likelihood need to keep track of site integrals:
  - minimising "unnormalised KL":

$$\mathbf{KL}[p \| q] = \int dx\, p(x) \log \frac{p(x)}{q(x)} + \int dx\, \big(q(x) - p(x)\big)$$

  incorporates normaliser into each $\tilde{f}$ (match zeroth moment, along with suff stats).

as well as the overall normaliser of $\prod_i \tilde{f}_i(\mathcal{Y}_i)$.

Alpha divergences and Power EP

- Alpha divergences $D_\alpha[p\|q] = \frac{1}{\alpha(1-\alpha)} \int dx\, \alpha p(x) + (1-\alpha)q(x) - p(x)^\alpha q(x)^{1-\alpha}$

Alpha divergences and Power EP

- Alpha divergences $D_\alpha[p\|q] = \frac{1}{\alpha(1-\alpha)} \int dx\, \alpha p(x) + (1-\alpha)q(x) - p(x)^\alpha q(x)^{1-\alpha}$

$$D_{-1}[p\|q] = \frac{1}{2} \int dx\, \frac{(p(x)-q(x))^2}{p(x)}$$

$$\lim_{\alpha\to 0} D_\alpha[p\|q] = \mathbf{KL}[q\|p]$$

Note: $\lim_{\alpha\to 0} \frac{(p(x)/q(x))^\alpha}{\alpha} = \log \frac{p(x)}{q(x)}$

$$D_{\frac{1}{2}}[p\|q] = 2 \int dx\, (p(x)^{\frac{1}{2}} - q(x)^{\frac{1}{2}})^2$$

$$\lim_{\alpha\to 1} D_\alpha[p\|q] = \mathbf{KL}[p\|q]$$

$$D_2[p\|q] = \frac{1}{2} \int dx\, \frac{(p(x)-q(x))^2}{q(x)}$$

Alpha divergences and Power EP

- Alpha divergences $D_\alpha[p\|q] = \frac{1}{\alpha(1-\alpha)} \int dx\, \alpha p(x) + (1-\alpha) q(x) - p(x)^\alpha q(x)^{1-\alpha}$

$$D_{-1}[p\|q] = \frac{1}{2} \int dx \frac{(p(x)-q(x))^2}{p(x)}$$

$$\lim_{\alpha\to 0} D_\alpha[p\|q] = \mathbf{KL}[q\|p] \qquad \text{Note: } \lim_{\alpha\to 0} \frac{(p(x)/q(x))^\alpha}{\alpha} = \log \frac{p(x)}{q(x)}$$

$$D_{\frac{1}{2}}[p\|q] = 2 \int dx\, (p(x)^{\frac{1}{2}} - q(x)^{\frac{1}{2}})^2$$

$$\lim_{\alpha\to 1} D_\alpha[p\|q] = \mathbf{KL}[p\|q]$$

$$D_2[p\|q] = \frac{1}{2} \int dx \frac{(p(x)-q(x))^2}{q(x)}$$

- Local (EP) minimisation gives fixed-point updates that blend messages (to power α) with previous site approximations.

$$\tilde{f}_i^{\text{new}} = \operatorname*{argmin}_{f\in\{\tilde{f}\}} \mathbf{KL}\left[f_i(\mathcal{Y}_i)^\alpha \tilde{f}_i(\mathcal{Y}_i)^{1-\alpha} q_{\neg i}(\mathcal{Y}) \middle\| f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y})\right]$$

Alpha divergences and Power EP

- Alpha divergences $D_\alpha[p\|q] = \frac{1}{\alpha(1-\alpha)} \int dx\, \alpha p(x) + (1-\alpha)q(x) - p(x)^\alpha q(x)^{1-\alpha}$

$$D_{-1}[p\|q] = \frac{1}{2}\int dx\, \frac{(p(x)-q(x))^2}{p(x)}$$

$$\lim_{\alpha\to 0} D_\alpha[p\|q] = \mathbf{KL}[q\|p] \qquad \text{Note: } \lim_{\alpha\to 0}\frac{(p(x)/q(x))^\alpha}{\alpha} = \log\frac{p(x)}{q(x)}$$

$$D_{\frac{1}{2}}[p\|q] = 2\int dx\, (p(x)^{\frac{1}{2}} - q(x)^{\frac{1}{2}})^2$$

$$\lim_{\alpha\to 1} D_\alpha[p\|q] = \mathbf{KL}[p\|q]$$

$$D_2[p\|q] = \frac{1}{2}\int dx\, \frac{(p(x)-q(x))^2}{q(x)}$$

- Local (EP) minimisation gives fixed-point updates that blend messages (to power α) with previous site approximations.

$$\tilde{f}_i^{\text{new}} = \operatorname*{argmin}_{f\in\{\tilde{f}\}} \mathbf{KL}\left[f_i(\mathcal{Y}_i)^\alpha \tilde{f}_i(\mathcal{Y}_i)^{1-\alpha} q_{\neg i}(\mathcal{Y}) \middle\| f(\mathcal{Y}_i) q_{\neg i}(\mathcal{Y})\right]$$

- Small changes (for $\alpha < 1$) lead to more stable updates, and more reliable convergence.