Kernel-Based Just-In-Time Learning for Passing Expectation Propagation Messages

27 April 2015

Wittawat Jitkrittum[1], Arthur Gretton[1], Nicolas Heess, S. M. Ali Eslami, Balaji Lakshminarayanan[1], Dino Sejdinovic[2], and Zoltán Szabó[1]

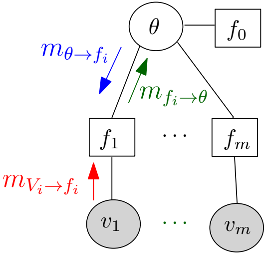
Gatsby Unit, UCL[1]
University of Oxford[2]

Gatsby Research Talk

Outline

Outline

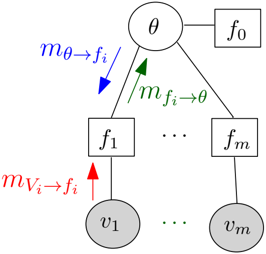
- Model: $p(\{v_i\}_i, \theta) \propto f_0(\theta) \prod_i f(v_i|\theta)$
- **Inference:** Find posterior of θ given observations $\{v_i\}_i$.
- **EP posterior:** $p(\theta|\{v_i\}_i) \propto f_0(\theta) \prod_{j=1}^{m} m_{f_j \to \theta}(\theta)$



$$m_{f_i \to \theta}(\theta) = \frac{\text{proj}\left[\int f(v'|\theta) m_{V_i \to f_i}(v') m_{\theta \to f_i}(\theta)\, dv'\right]}{m_{\theta \to f_i}(\theta)}$$

- $m_{V_i \to f_i}(v') = \delta_{v_i}(v')$
- $\text{proj}[r]$ projects r onto exponential family.
- Cavity distribution $m_{\theta \to f_i}(\theta) \propto \prod_{j \neq i} m_{f_j \to \theta}(\theta)$ gives context of what other $\{v_j\}_{j \neq i}$ say about θ.

Inference and EP



- Model: $p(\{v_i\}_i, \theta) \propto f_0(\theta) \prod_i f(v_i|\theta)$
- **Inference:** Find posterior of θ given observations $\{v_i\}_i$.
- **EP posterior:** $p(\theta|\{v_i\}_i) \propto f_0(\theta) \prod_{j=1}^{m} m_{f_j\to\theta}(\theta)$

$$m_{f_i\to\theta}(\theta) = \frac{\mathrm{proj}\left[\int f(v'|\theta) m_{V_i\to f_i}(v') m_{\theta\to f_i}(\theta)\, dv'\right]}{m_{\theta\to f_i}(\theta)}$$

- $m_{V_i\to f_i}(v') = \delta_{v_i}(v')$
- $\mathrm{proj}[r]$ projects r onto exponential family.
- Cavity distribution $m_{\theta\to f_i}(\theta) \propto \prod_{j\neq i} m_{f_j\to\theta}(\theta)$ gives context of what other $\{v_j\}_{j\neq i}$ say about θ.

General EP Outgoing Messages

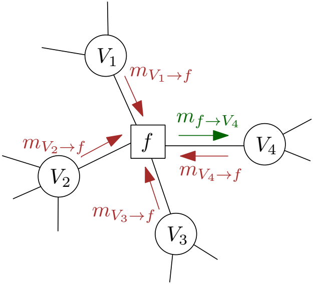
set of c variables connected to f

projected message

$$m_{f\to V_i}(v_i) = \frac{\text{proj}\left[\int d\mathcal{V}\backslash\{v_i\}\, f(\mathcal{V}) \prod_{j=1}^{c} m_{V_j\to f}(v_j)\right]}{m_{V_i\to f}(v_i)} := \frac{q_{f\to V_i}(v_i)}{m_{V_i\to f}(v_i)}$$

$\text{proj}[r_{f\to V_i}] := \arg\min_{q\in \text{ExpFam}} \text{KL}\left[r_{f\to V_i} \,\|\, q\right]$
(projection onto exponential family)

incoming message from V_j



- Expensive integral.
- **Goal:** Learn an uncertainty aware message operator (regression function)

$$\left[m_{V_j\to f}\right]_{j=1}^{c} \mapsto q_{f\to V_i}.$$

- If uncertain, ask the oracle to get $q_{f\to V_i}$, and update itself online.

General EP Outgoing Messages

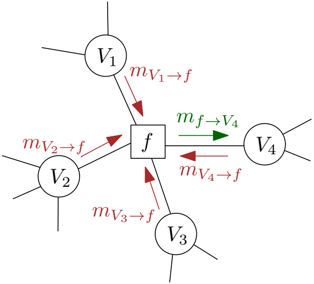
set of c variables connected to f

projected message

$$m_{f\to V_i}(v_i) = \frac{\text{proj}\left[\int d\mathcal{V}\backslash\{v_i\}\, f(\mathcal{V}) \prod_{j=1}^{c} m_{V_j\to f}(v_j)\right]}{m_{V_i\to f}(v_i)} := \frac{q_{f\to V_i}(v_i)}{m_{V_i\to f}(v_i)}$$

$\text{proj}[r_{f\to V_i}] := \arg\min_{q\in\text{ExpFam}} \text{KL}\left[r_{f\to V_i} \,\|\, q\right]$
(projection onto exponential family)

incoming message from V_j



- Expensive integral.
- **Goal:** Learn an <u>uncertainty aware</u> message operator (regression function)

$$\left[m_{V_j\to f}\right]_{j=1}^{c} \mapsto q_{f\to V_i}.$$

- If uncertain, ask the oracle to get $q_{f\to V_i}$, and update itself online.

Outline

Just-In-Time (JIT) Learning to Infer

Propose kernel-based JIT learning (KJIT) to send EP messages.

- Faster with same inference quality.
- Automatic random feature representation of input messages.
- Generic solution for <u>any</u> factor f that can be sampled.
- Learned operator reusable.

Projection onto Exponential Family

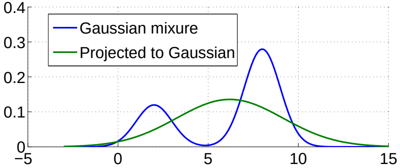
- $q \in$ ExpFam:

$$q(v|\eta) = \exp\left(\eta^\top u(v) - A(\eta)\right).$$

- $q = \text{proj}\,[r] = \arg\min_{q \in \text{ExpFam}} \text{KL}[r \,\|\, q]$ satisfies

$$\mathbb{E}_r[u(v)] = \mathbb{E}_q[u(v)] \quad \text{(moment matching)}.$$

- Need only $\mathbb{E}_r[u(v)]$ to form q.
- Computed with importance sampling.



- If q is a Gaussian, $u(v) = (v, v^2)^\top$.
- $\mathbb{E}_r[u(v)]$ = regression target.

Gaussian Process Regression

- X: N training tuples of input messages $\left[m_{V_j \to f}\right]_{j=1}^{c}$.
- Y: one coordinate of $\mathbb{E}_r[u(v)]$.
- σ_y^2: noise variance.
- κ: kernel on input messages.

GP regression:

$$y^* \mid X, Y, x^* \sim \mathcal{N}\Big(y^* \,\Big|\, \kappa(x^*, X)\left(\kappa(X, X) + \sigma_y^2 I\right)^{-1} Y^\top, \\ \kappa(x^*, x^*) - \kappa(x^*, X)\left(\kappa(X, X) + \sigma_y^2 I\right)^{-1} \kappa(X, x^*)\Big).$$

- Not suitable for online learning because size of $\kappa(X, X)$ grows.

Gaussian Process in Primal Form

- Let $x_n := \left[m^{(n)}_{V_j \to f}\right]_{j=1}^{c}$ (n^{th} tuple of input messages)
- **Idea**: Approximate $\kappa(x_m, x_n) \approx \hat{\psi}(x_m)^\top \hat{\psi}(x_n)$ where $\hat{\psi}(x_n) \in \mathbb{R}^D$ (random features).
- Let $\mathsf{x}_n := \hat{\psi}(x_n)$.
- Input: $\mathsf{X} = (\mathsf{x}_1 | \cdots | \mathsf{x}_N) \in \mathbb{R}^{D \times N}$

GP regression becomes

$$
\begin{aligned}
y^* \mid \mathsf{X}, Y, x^* &\sim \mathcal{N}(y^* \mid \mu_w^\top \mathsf{x}^*,\ \mathsf{x}^{*\top} \Sigma_w \mathsf{x}^* + \sigma_y^2), \\
\Sigma_w &= \left(\mathsf{X}\mathsf{X}^\top \sigma_y^{-2} + \sigma_0^{-2} I\right)^{-1} \in \mathbb{R}^{D \times D}, \\
\mu_w &= \Sigma_w \mathsf{X} Y^\top \sigma_y^{-2} \ \in \mathbb{R}^{D}.
\end{aligned}
$$

where σ_0^2 = prior variance.

- Solutions Σ_w, μ_w do not grow with N.

Online Update

- Need to maintain Σ_w and μ_w.
- $\cdot^{[N]}$:= quantity constructed from N samples.
- By Sherman-Morrison formula,

$$\Sigma_w^{[N+1]} = \Sigma_w^{[N]} - \frac{\Sigma_w^{[N]} \mathsf{x}_{N+1} \mathsf{x}_{N+1}^\top \Sigma_w^{[N]} \sigma_y^{-2}}{1 + \mathsf{x}_{N+1}^\top \Sigma_w^{[N]} \mathsf{x}_{N+1} \sigma_y^{-2}}.$$

- For $\mu_w = \Sigma_w \mathsf{X} Y^\top \sigma_y^{-2}$,

$$\left(\mathsf{X} Y^\top\right)^{[N+1]} = \left(\mathsf{X} Y^\top + \mathsf{X}_{N+1} y_{N+1}\right) \in \mathbb{R}^D.$$

- Cheap updates as a function of previous solution.

κ: Gaussian Kernel on Mean Embeddings

- Product distribution of c incoming messages: $\mathsf{r} := \times_{l=1}^{c} r_l$, $\mathsf{s} := \times_{l=1}^{c} s_l$.
- Gaussian kernel on mean embeddings:

$$\kappa(\mathsf{r}, \mathsf{s}) = \exp\left(-\frac{\|\mu_{\mathsf{r}} - \mu_{\mathsf{s}}\|_{\mathcal{H}}^2}{2\gamma^2}\right)$$

  where $\mu_{\mathsf{r}} := \mathbb{E}_{a \sim \mathsf{r}} \varphi(a)$ (mean embedding of r).
- Two-stage random feature approximation:

$$\kappa(\mathsf{r}, \mathsf{s}) \overset{1^{st}}{\approx} \exp\left(-\frac{\|\hat{\phi}(\mathsf{r}) - \hat{\phi}(\mathsf{s})\|_{D_{\text{in}}}^2}{2\gamma^2}\right) \overset{2^{nd}}{\approx} \hat{\psi}(\mathsf{r})^\top \hat{\psi}(\mathsf{s}).$$

Approximating κ

$$\kappa(\mathsf{r},\mathsf{s}) = \exp\left(-\frac{1}{2\gamma^2}\langle\mu_\mathsf{r},\mu_\mathsf{r}\rangle + \frac{1}{\gamma^2}\langle\mu_\mathsf{r},\mu_\mathsf{s}\rangle - \frac{1}{2\gamma^2}\langle\mu_\mathsf{s},\mu_\mathsf{s}\rangle\right).$$

Consider $\langle\mu_\mathsf{r},\mu_\mathsf{s}\rangle$:

$$\begin{aligned}
\langle\mu_\mathsf{r},\mu_\mathsf{s}\rangle &= \mathbb{E}_{a\sim\mathsf{r}}\mathbb{E}_{b\sim\mathsf{s}}\langle\varphi(a),\varphi(b)\rangle \\
&= \mathbb{E}_{a\sim\mathsf{r}}\mathbb{E}_{b\sim\mathsf{s}}k(a,b) \\
\text{(approximate } k\text{)} &\approx \mathbb{E}_{a\sim\mathsf{r}}\mathbb{E}_{b\sim\mathsf{s}}\hat{\varphi}(a)^\top\hat{\varphi}(b) \\
&= \mathbb{E}_{a\sim\mathsf{r}}\hat{\varphi}(a)^\top\mathbb{E}_{b\sim\mathsf{s}}\hat{\varphi}(b) \\
&:= \hat{\phi}(\mathsf{r})^\top\hat{\phi}(\mathsf{s}).
\end{aligned}$$

- k: Gaussian kernel.
- Same random feature approximation [Rahimi and Recht, 2007] twice.

$$\kappa(\mathsf{r},\mathsf{s}) \overset{1^{st}}{\approx} \underbrace{\exp\left(-\frac{\|\hat{\phi}(\mathsf{r})-\hat{\phi}(\mathsf{s})\|^2_{D_{\text{in}}}}{2\gamma^2}\right)}_{\text{finite-dimensional Gaussian kernel}} \overset{2^{nd}}{\approx} \hat{\psi}(\mathsf{r})^\top\hat{\psi}(\mathsf{s}).$$

Random Fourier Features [Rahimi and Recht, 2007] I

Bochner's theorem

A continuous, translation-invariant kernel $k(a, b) = k(a - b)$ on $\mathbb{R}^m$ is positive definite iff

$$k(a - b) = \int e^{i\omega^\top (a-b)} \hat{k}(\omega) \, \mathrm{d}\omega$$

for some probability measure $\hat{k}(\omega)$ (finite non-negative measure).

Goal: $k(a - b) \approx \hat{\varphi}(a)^\top \hat{\varphi}(b)$.

$$k(a-b) = \mathbb{E}_\omega \cos(\omega^\top (a-b)) + i \underbrace{\mathbb{E}_\omega \sin(\omega^\top (a-b))}_{=0}$$

$$\stackrel{(a)}{=} \mathbb{E}_{c \sim U[0,2\pi]} \mathbb{E}_\omega \left[2\cos(\omega^\top a + c) \cos(\omega^\top b + c) \right].$$

(a): $2\cos(\omega^\top a + c)\cos(\omega^\top b + c) = \cos(\omega^\top (a-b)) + \underbrace{\cos(\omega^\top (a+b) + 2c)}_{\mathbb{E}_{c \sim U[0,2\pi]} \text{ gives } 0}$

where we used $2\cos(x)\cos(y) = \cos(x - y) + \cos(x + y)$.

Random Fourier Features [Rahimi and Recht, 2007] I

Bochner's theorem

A continuous, translation-invariant kernel $k(a, b) = k(a - b)$ on $\mathbb{R}^m$ is positive definite iff

$$k(a - b) = \int e^{i\omega^\top (a-b)} \hat{k}(\omega) \, \mathrm{d}\omega$$

for some probability measure $\hat{k}(\omega)$ (finite non-negative measure).

Goal: $k(a - b) \approx \hat{\varphi}(a)^\top \hat{\varphi}(b)$.

$$k(a - b) = \mathbb{E}_\omega \cos(\omega^\top (a - b)) + i \underbrace{\mathbb{E}_\omega \sin(\omega^\top (a - b))}_{=0}$$

$$\overset{(a)}{=} \mathbb{E}_{c \sim U[0,2\pi]} \mathbb{E}_\omega \left[2 \cos(\omega^\top a + c) \cos(\omega^\top b + c) \right].$$

(a): $2 \cos(\omega^\top a + c) \cos(\omega^\top b + c) = \cos(\omega^\top (a - b)) + \underbrace{\cos(\omega^\top (a + b) + 2c)}_{\mathbb{E}_{c \sim U[0,2\pi]} \text{ gives } 0}$

where we used $2 \cos(x) \cos(y) = \cos(x - y) + \cos(x + y)$.

Random Fourier Features [Rahimi and Recht, 2007] II

$$k(a-b) = \mathbb{E}_{c\sim U[0,2\pi]}\mathbb{E}_{\omega}\left[2\cos(\omega^\top a + c)\cos(\omega^\top b + c)\right]$$

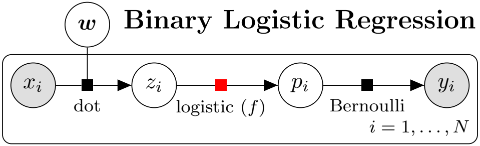
(empirical average) $\approx \hat{\varphi}(a)^\top \hat{\varphi}(b).$

Random features $\hat{\varphi}(a) \in \mathbb{R}^D$ such that $k(a-b) \approx \hat{\varphi}(a)^\top \hat{\varphi}(b)$:

1. Draw i.i.d. $\{\omega_i\}_{i=1}^D \sim \hat{k}(\omega)$.
2. Draw i.i.d. $\{c_i\}_{i=1}^D \sim U[0, 2\pi]$ to correct bias.
3. $\hat{\varphi}(a) = \sqrt{\frac{2}{D}}\left[\cos\left(\omega_1^\top a + c_1\right), \ldots, \cos\left(\omega_D^\top a + c_D\right)\right]^\top \in \mathbb{R}^D$

Outline

Experiment 1: Message Prediction



$$\textcolor{red}{f}(p|z) = \delta_p\left(\frac{1}{1+\exp(-z)}\right)$$

- 2 incoming messages:

$$m_{z\to f}(z) = \text{Gaussian}(z)$$
$$m_{p\to f}(p) = \text{Beta}(p)$$

- Predict

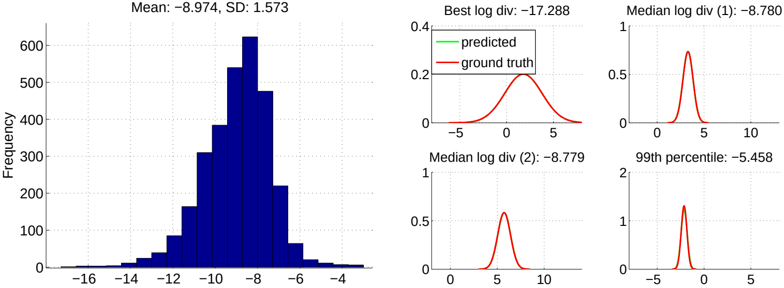
$$q_{f\to z}(z) = \text{proj}\left[\int \textcolor{red}{f}(p|z) m_{p\to f}(p) m_{z\to f}(z)\, \mathrm{d}p\right] = \text{Gaussian}(z),$$

from $(m_{z\to f}, m_{p\to f})$.

Batch Learning Result

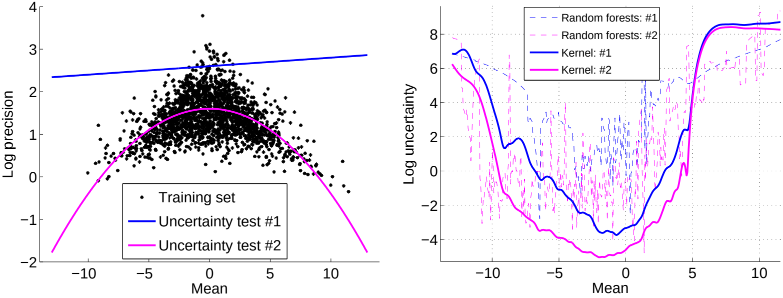
- Batch train on messages collected from 20 EP runs on toy data (binary logistic regression).
- Train/test: 5000/3000.
- Report

$$\log \mathrm{KL}\left[\text{true } q_{f\to z} \,\|\, \text{predicted } \hat{q}_{f\to z}\right].$$



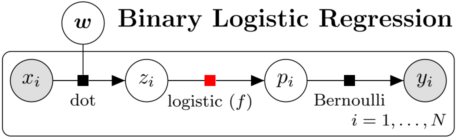
Experiment 2: Uncertainty Estimates

- Same training set as before.

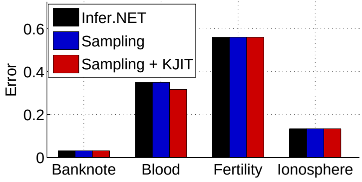


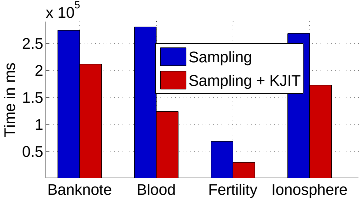
- **Left:** Parameters of $m_{z \to f}$.
- **Right:** KJIT gives smoother uncertainty estimates (predictive variance).
- Fix beta messages $m_{p \to f}$ during testing.

Experiment 3: EP on Real Data



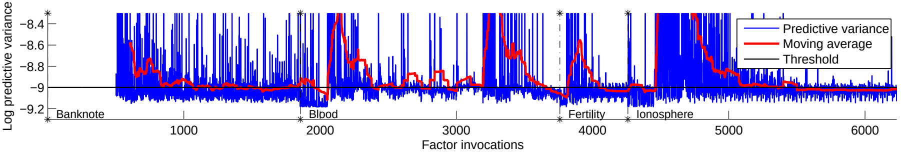
- Sequentially present 4 real datasets to the operator to JIT learn.
- If predictive variance $>$ threshold, ask oracle.



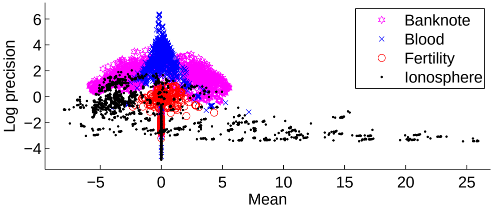


- **Left:** Binary classification error with learned posterior $\boldsymbol{w}$.
  - Infer.NET = handcrafted operator.
- **Right:** EP runtime.

Changes in Input Message Distribution



- Initial silent period = parameter selection + mini-batch training.
- $*$ = start of a new problem.
- Sharp rises after $*$ indicate ability to detect distribution (problem) change.



- $\leftarrow$ Diverse distributions of $m_{z \to f} = \text{Gaussian}(z)$.

Conclusion

- Proposed KJIT, a kernel-based message operator.
- KJIT learns to send messages online during EP.
- Automatic representation of input messages.
- Uncertainty aware.
- Faster than ordinary EP with same inference quality.

More info:

- Paper: http://arxiv.org/abs/1503.02551
- Code: http://wittawat.com

References I

Rahimi, A. and Recht, B. (2007).
Random features for large-scale kernel machines.
In NIPS, pages 1177–1184.

Importance Sampling Oracle

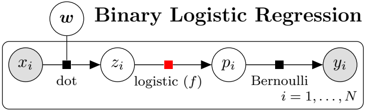
$$q_{f\to V_i}(v_i) \quad = \quad \text{proj}\left[\int f(v_1|v_{2:c})\prod_{j=1}^{c} m_{V_j\to f}(v_j)\,\mathrm{d}\mathcal{V}\backslash\{v_i\}\right] = \text{proj}[r]$$

- To compute $q_{f\to V_i}(v_i)$,

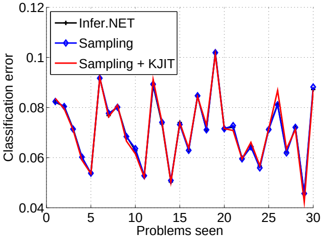
$$\begin{aligned}
\mathbb{E}_r[u(v_i)] &= \int u(v_i) f(v_1|v_{2:c})\prod_{j=1}^{c} m_{V_j\to f}(v_j)\,\mathrm{d}\mathcal{V} \\
&= \int u(v_i)\frac{\prod_{j=1}^{c} m_{V_j\to f}(v_j)}{s(v_{2:c})} f(v_1|v_{2:c})s(v_{2:c})\,\mathrm{d}\mathcal{V} \\
&\approx \frac{1}{K}\sum_{k=1}^{K} u(v_i^{(k)})\frac{\prod_{j=1}^{c} m_{V_j\to f}(v_j^{(k)})}{s(v_{2:c}^{(k)})}
\end{aligned}$$

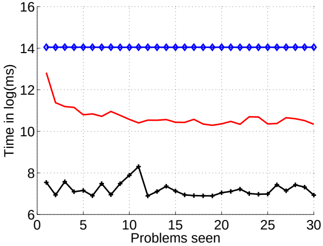
  where $\left\{v^{(k)}\right\}_k \sim f(v_1|v_{2:c})s(v_{2:c})$ and $s(v_{2:c})$ is a proposal distribution.
- Only need the ability to sample from f.

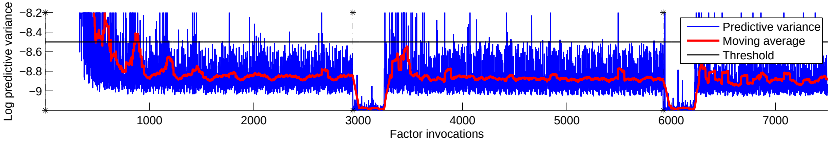
Binary Logistic Regression: Toy Data



- Fix true $\boldsymbol{w}$. Sequentially present 30 problems.
- Generate $\{x_i, y_i\}_{i=1}^{300}$ for each.





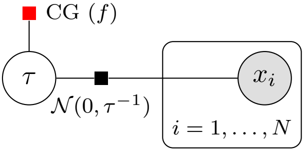


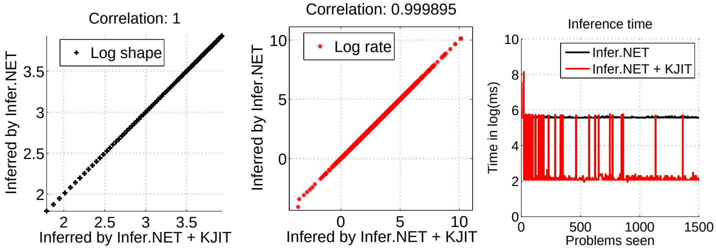
- As good as handcrafted factor; much faster.

Experiment: Compound Gamma Factor

- **Goal:** Infer posterior precision τ of $x \sim \mathcal{N}(x; 0, \tau^{-1})$ from observations $\{x_i\}_{i=1}^{N}$.

$$r_2 \sim \mathrm{Gamma}(r_2; s_1, r_1)$$
$$\tau \sim \mathrm{Gamma}(\tau; s_2, r_2)$$
$$(s_1, r_1, s_2) = (1, 1, 1)$$





- **Infer.NET + KJIT** = KJIT with handcrafted factor oracle.

Performance of Different Kernels

- EPP := expected product kernel
- RF := random feature
- IChol := incomplete Cholesky to approximate the gram matrix.
- MV kernel = Gaussian kernel on mean and variance of messages.

	mean log KL	**SD**
RF + MV Kernel	-6.9554	1.6726
RF + EP on joint embeddings	-2.7765	1.8261
RF + Sum of EPPs	-1.0518	1.9315
RF + Product of EPPs	-2.641	1.645
RF + Gauss. kernel on joint embeddings	-8.9740	1.5731
IChol + sum of Gauss. kernel on embeddings	-2.751	2.8382
IChol + Gauss. kernel on joint embeddings	-8.7144	1.6864

- Dataset = messages collected from 20 EP runs on toy data of binary logistic regression.

Three Ways to Minimize KL

1. Simple. Local. Treat each factor independently.

$$\tilde{f}_i = \arg \min_{\tilde{f} \in \text{ExpFam}} KL\left[f_i(\mathcal{X}_i|\theta) \,\middle\|\, \tilde{f}(\theta)\right]$$

2. Globally accurate. But intractable.

$$\begin{aligned} q^*(\theta) &= \arg \min_{q \in \text{ExpFam}} KL\left[f_0(\theta) \prod_{i=1}^{m} f_i(\mathcal{X}_i|\theta) \,\middle\|\, \tilde{f}_0(\theta) \prod_{i=1}^{m} \tilde{f}_i(\mathcal{X}_i|\theta)\right] \\ &= \arg \min_{q \in \text{ExpFam}} KL\left[p(\theta|\mathcal{X}) \,\middle\|\, q(\theta)\right] \end{aligned}$$

3. EP is in between the previous two. Iterative. Contextual.

$$\begin{aligned} q_i(\theta) &= \arg \min_{q \in \text{ExpFam}} KL\left[f_i(\mathcal{X}_i|\theta) \textstyle\prod_{j \neq i} \tilde{f}_j(\mathcal{X}_j|\theta) \,\middle\|\, q(\theta)\right] \\ &= \arg \min_{q \in \text{ExpFam}} KL\left[f_i(\mathcal{X}_i|\theta)\, q^{\setminus i}(\theta) \,\middle\|\, q(\theta)\right] := \text{proj}\left[f_i(\mathcal{X}_i|\theta)\, q^{\setminus i}(\theta)\right] \\ \tilde{f}_i &\propto q_i(\theta)/q^{\setminus i}(\theta) \end{aligned}$$

Posterior is constructed by $q^*(\theta) := f_0(\theta) \prod_{i=1}^{m} \tilde{f}_i(\mathcal{X}_i|\theta)$.

Two-Stage Random Features $\hat{\psi}$ for κ

Construction of two-stage random features for κ.

In: $\mathcal{F}(k)$: Fourier transform of k, D_{in}: #inner features, D_{out}: #outer features, k_{gauss}: Gaussian kernel on $\mathbb{R}^{D_{\text{in}}}$

Out: Random features $\hat{\psi}(\mathsf{r}) \in \mathbb{R}^{D_{\text{out}}}$

1: Sample $\{\omega_i\}_{i=1}^{D_{\text{in}}} \overset{i.i.d}{\sim} \mathcal{F}(k)$,

2: Sample $\{b_i\}_{i=1}^{D_{\text{in}}} \overset{i.i.d}{\sim} U[0, 2\pi]$.

3: $\hat{\phi}(\mathsf{r}) = \sqrt{\frac{2}{D_{\text{in}}}} \left(\mathbb{E}_{\mathsf{x}\sim\mathsf{r}} \cos(\omega_i^\top x + b_i)\right)_{i=1}^{D_{\text{in}}} \in \mathbb{R}^{D_{\text{in}}}$

4: Sample $\{\nu_i\}_{i=1}^{D_{\text{out}}} \overset{i.i.d}{\sim} \mathcal{F}(k_{\text{gauss}}(\gamma^2))$

5: Sample $\{c_i\}_{i=1}^{D_{\text{out}}} \overset{i.i.d}{\sim} U[0, 2\pi]$.

6: $\hat{\psi}(\mathsf{r}) = \sqrt{\frac{2}{D_{\text{out}}}} \left(\cos(\nu_i^\top \hat{\phi}(\mathsf{r}) + c_i)\right)_{i=1}^{D_{\text{out}}} \in \mathbb{R}^{D_{\text{out}}}$

Factor-based View of EP

- $q(\theta) = \mathcal{N}(\theta|m_0, v_0)$ (assume $\{\tilde{f}_i\}_i$ are Gaussian)
- $\tilde{f}_i(\theta) = 1$ for $i = 1, \ldots, m$.
- Repeat EP iterations until convergence (several passes over $1, \ldots, m$)
  - for each factor $i = 0 \ldots, m$
    - **Deletion**: $q^{\setminus i}(\theta) \propto q(\theta)/\tilde{f}_i(\theta) = \prod_{j \neq i} \tilde{f}_j(\theta)$
    - **Inclusion**: $q(\theta) = \text{proj}\left[f_i(\mathcal{X}_i|\theta)q^{\setminus i}(\theta)\right]$
    - **Update**: $\tilde{f}_i(\theta) = q(\theta)/q^{\setminus i}(\theta)$
- $q^*(\theta) = f_0(\theta) \prod_{i=1}^{m} \tilde{f}_i(\mathcal{X}_i|\theta)$

- $q^{\setminus i}(\theta) \propto \prod_{j \neq i} \tilde{f}_j(\theta)$ is called the cavity distribution giving a context of what others $\left(\left\{\tilde{f}_j(\theta)\right\}_{j \neq i}\right)$ say about θ.
- If $r \in$ ExpFam, then $r = \text{proj}\,[r]$.
- ExpFam is closed under multiplication and division.

Random Features for Expected Product Kernels I

$$k_{\mathsf{pro}}(p,q) = \langle \mu_p, \mu_q \rangle_{\mathcal{H}} = \mathbb{E}_{p(x)}\mathbb{E}_{q(y)} k_{\mathsf{gauss}}(x,y)$$

With random features,

$$\begin{aligned}
\mathbb{E}_p\mathbb{E}_q k_{\mathsf{gauss}}(x,y) &\approx \mathbb{E}_{p(x)}\mathbb{E}_{q(y)} \hat{\phi}(x)^\top \hat{\phi}(y) \\
&= \mathbb{E}_p\mathbb{E}_q \frac{2}{D}\sum_{i=1}^{D} \cos\left(\omega_i^\top x + b_i\right)\cos\left(\omega_i^\top y + b_i\right) \\
&= \frac{2}{D}\sum_{i=1}^{D} \mathbb{E}_{p(x)} \cos\left(\omega_i^\top x + b_i\right) \mathbb{E}_{q(y)} \cos\left(\omega_i^\top y + b_i\right)
\end{aligned}$$

Assume $p(x) = \mathcal{N}(x; M_p, V_p)$ and $q(y) = \mathcal{N}(y; M_q, V_q)$,

$$\mathbb{E}_{p(x)} \cos\left(\omega_i^\top x + b_i\right) = \cos(\omega_i^\top M_p + b_i) \exp\left(-\frac{1}{2}\omega_i^\top V_p \omega_i\right)$$

Random Features for Expected Product Kernels II

$$\hat{\varphi}(p) = \sqrt{\frac{2}{D}} \begin{pmatrix} \cos(\omega_1^\top M_p + b_1) \exp\left(-\frac{1}{2}\omega_1^\top V_p w_1\right) \\ \vdots \\ \cos(w_D^\top M_p + b_D) \exp\left(-\frac{1}{2} w_D^\top V_p w_D\right) \end{pmatrix}.$$

Assume $k_{\mathsf{gauss}}(x,y) = \exp\left(-\frac{1}{2}(x-y)^\top \Sigma^{-1}(x-y)\right)$ where Σ is the kernel parameter, and Gaussian p, q,

$$\mathbb{E}_p \mathbb{E}_q k_{\mathsf{gauss}}(x,y) \overset{\text{(exact)}}{=} \sqrt{\frac{\det(D_{pq})}{\det(\Sigma^{-1})}} \exp\left(-\frac{1}{2}(M_p - M_q)^\top D_{pq}(M_p - M_q)\right)$$

$$D_{pq} := (V_p + V_q + \Sigma)^{-1}$$