Appendix of:
Inverse Reinforcement Learning via Deep Gaussian Process

1 Background: Inverse Reinforcement Learning and DGP-IRL

The Markov Decision Process (MDP) is characterized by \{S, A, T, \gamma, r\}, which represents the state space, action space, transition model, discount factor, and reward function, respectively.

The IRL task is to find the reward function \(r^*\) such that the induced optimal policy matches the demonstrations, given \{S, A, T, \gamma\} and \(M = \{\zeta_1, \ldots, \zeta_h\}\), where \(\zeta_i = \{(s_{i,t}, a_{i,t}) : t = 1, \ldots, T_i\}\) is the demonstration trajectory, consisting of state-action pairs.

Deep Gaussian process for inverse reinforcement learning (DGP-IRL) extends the deep Gaussian process (deep GP) framework to the IRL domain, as shown in Fig. 1. DGP-IRL learns an abstract representation that reveals the reward structure by warping the original feature space through the latent layers, \(D, B\).

![Figure 1: The proposed deep GP model for IRL, where latent Gaussian processes are introduced to learn a representation of the world for the latent reward \(r\). The rewards are provided to the reinforcement learning (RL) engine to generate a set of observable trajectories \(M\).](image)

For a set of observed trajectories \(M\), our objective is to optimize the corresponding marginalized log-likelihood given the states in the world as represented by \(X\):

\[
\log p(M|X) = \log \int p(M|r)p(r|D)p(D|B)p(B|X)d(r, D, B)
\]

(1)

where the integration is with respect to the latent layers, including the reward vector \(r\). As introduced in the main paper, \(d^m \in \mathbb{R}^n\) is the \(m\)-th column of the latent layer \(D = [d^1 \ldots d^m]\), and similarly for \(B = [b^1 \ldots b^m]\):

\[
p(M|r) = \sum_{i=1}^h \sum_{t=1}^T (Q(s_{i,t}, a_{i,t}; r) - V(s_{i,t}; r))
\]

(2)

\[
p(r|D) = \mathcal{N}(r|0, K_{DD})
\]

(3)

\[
p(D|B) = \prod_{m=1}^m \mathcal{N}(d^m|b^m, \lambda^{-1}I)
\]

(4)

\[
p(B|X) = \prod_{m=1}^m \mathcal{N}(b^m|0, K_{XX})
\]

(5)

where \(p(M|r)\) represents the reinforcement learning term, given by:

\[
\log p(M|r) = \sum_i \sum_t (Q(s_{i,t}, a_{i,t}; r) - V(s_{i,t}; r))
\]

(6)

\[
= \sum_i \sum_t \left( r_{s_{i,t}, a_{i,t}} - V(s_{i,t}; r) + \sum_{s'} \gamma T_{s,t}^{s_{i,t}, a_{i,t}} V(s'; r) \right)
\]

(7)

The Q-value \(Q(s_{i,t}, a_{i,t}; r)\) used above is a measure of how desirable is the corresponding state-action pair \((s_{i,t}, a_{i,t})\) under rewards \(r\) for all the world states, and is defined by:

\[
Q(s_{i,t}, a_{i,t}; r) = r_{s_{i,t}, a_{i,t}} + \sum_{s'} \gamma T_{s,t}^{s_{i,t}, a_{i,t}} V(s'; r)
\]
where $r_{s_i,t,a_i,t} = r(s_{i,t}, a_{i,t}) \in \mathbb{R}$ is the reward for $(s_{i,t}, a_{i,t})$, $\gamma$ is the discount factor, $T_{s'_i,t,a_i,t} = P(s'_i|s_{i,t}, a_{i,t})$ is the transition probability by the transition model, and $V(s_{i,t}; r)$ is the value associated with state $s_{i,t}$, obtained by the modified Bellman backup operator:

$$V(s_{i,t}; r) = \log \sum_{a \in A} \exp \left( r_{s_i,t,a_i,t} + \sum_{s'} \gamma T_{s'_i,t,a_i,t} V(s'; r) \right)$$

where we apply a soft-max function $V(s_{i,t}; r) = \log \sum_{a \in A} \exp (Q(s_{i,t}, a_{i,t}; r))$ for the Q-values with all possible actions $a \in A$. The value function $V(s; r)$ for state $s$ can be obtained by repeatedly applying the above Bellman backup operator. For simplicity of notations, we use $V(s_{i,t}; r), Q(s_{i,t}, a_{i,t}; r)$ to denote the solution after Bellman backup operators, unlike some literature that uses $V^*(s_{i,t}; r), Q^*(s_{i,t}, a_{i,t}; r)$ to denote the difference. Detailed derivation of the above relationships can be found in [4].

2 Variational Lower Bound for DGP-IRL

It is intractable to perform the integration as in (1) for the marginal log-likelihood. In addition to $p(M|r)$, which involves the latent variable $r$ in a way which requires Q-value iterations, the term $p(r|D) = \mathcal{N}(r|0, K_{DD})$ has a nonlinear dependency on $D$ in the kernel matrix.

To tackle this issue, we introduce inducing outputs $f$, $V$ and their corresponding inputs $Z$, $W$, as shown in Fig. 2. The resulting model follows the main paper:

$$p(M|r) = \sum_{i=1}^h \sum_{t=1}^T (Q(s_{i,t}, a_{i,t}; r) - V(s_{i,t}; r))$$

(8)

$$p(r|f, D, Z) = \mathcal{N}(r|K_{DZ}K_{ZZ}^{-1}f, 0)$$

(9)

$$p(f|Z) = \mathcal{N}(f|0, K_{ZZ})$$

(10)

$$p(D|B) = \prod_{m=1}^{m_1} \mathcal{N}(d^m|b^m, \lambda^{-1}I)$$

(11)

$$p(B|V, X, W) = \prod_{m=1}^{m_1} \mathcal{N}(b^m|K_{XW}K_{WW}^{-1}v^m, \Sigma_B)$$

(12)

We also design the variation distribution as illustrated in the main paper:

$$Q = q(f)q(D)p(B|V, X)q(V), \text{ with:}$$

$$q(f) = \delta(f - \hat{f})$$

$$q(D) = \prod_{m=1}^{m_1} \delta(d^m - K_{XW}K_{WW}^{-1}\tilde{v}^m)$$

$$q(V) = \prod_{m=1}^{m_1} \mathcal{N}(v^m|\tilde{v}^m, \Sigma_B)$$
where the variational distribution \( Q \) is to not be confused with the notation for Q-values, \( Q \). Using the above distribution \( Q \), we can derive the variational lower bound as follows:

\[
\log p(M|X, Z, W) = \log \int p(M, r, f, V, D, B|Z, W, X)d(r, f, V, D, B) \tag{13}
\]

\[
= \log \int \frac{p(M|r)p(r|f, D, Z)p(f|Z)p(D|B)p(B|V, W, X)p(V|W)d(r, f, V, D, B)}{p(M|K_DZK_{ZZ}^{-1}f)} \tag{14}
\]

\[
\geq \int q(f)q(D)p(B|V, W, X)q(V) \log \frac{p(M|K_DZK_{ZZ}^{-1}f)p(f|Z)p(D|B)p(V|W)}{q(f)q(D)q(V)} \tag{15}
\]

\[
= \log p(M|K_DZK_{ZZ}^{-1}f) + \log p(f = \tilde{f}|Z) + \int q(V)q(D)p(B|V, W, X) \log \frac{p(D|B)p(V|W)}{q(V)}d(D, B, V) \tag{16}
\]

In the above derivation, the combination of \( p(M|r)p(r|f, D, Z) \) in (14) uses the deterministic training conditional (DTC) assumption \[2\], i.e., \( p(r|f, D, Z) = \delta(r - K_DZK_{ZZ}^{-1}f) \). (15) applies Jensen's inequality with the variational distribution \( Q \). (16) is a direct consequence of the choice of \( Q \), and \( D = [d^{1*} \cdots d^{m*}] \), with \( d^{m} = K_{XW}K_{WW}^{-1}v^{m} \).

**Utility 1 (Gaussian identities)** If the marginal and conditional Gaussian distributions for \( f \) and \( v \) are in the form:

\[
p(f|v) = N(f|M\mu_f + m, \Sigma_f)
p(v) = N(v|\mu_v, \Sigma_v)
\]

Then the marginal distribution of \( f \) is:

\[
p(f) = N(f|M\mu_v + m, \Sigma_f + M\Sigma_vM^T) \tag{17}
\]

Using the Gaussian identities, the derivation of \( \int q(V)p(B|V, W, X)dV \) is as follows:

\[
\int q(V)p(B|V, W, X)dV = \int \prod_{m=1}^{m_1} N(v_m^{*}|v_m^{*}, G_m^{*})N(b_m^*|K_{XW}K_{WW}^{-1}v_m^{*}, \Sigma_B)dV
\]

\[
= \prod_{m=1}^{m_1} N(b_m^*|K_{XW}K_{WW}^{-1}v_m^{*}, \Sigma_B + K_{XW}K_{WW}^{-1}G_m^*K_{WW}^{-1}K_{WX})
\]

Figure 2: Illustration of DGP-IRL with the inducing outputs \( f, V \) and the corresponding inputs \( Z, W \).
Therefore, we can obtain a closed form integration for the last term in (16) as follows:

\[
\int q(V)q(D)p(B|V, W, X)\log p(D|B)d(D, B, V)
\]

\[
= \int \left( \int q(V)p(B|V, W, X)dV \right) q(D)\log p(D|B)d(D, B)
\]

\[
= \int \prod_{m=1}^{m_1} \mathcal{N}(b^m|\tilde{b}^m, \Sigma_B^m) \log \prod_{m=1}^{m_1} \mathcal{N}(d^m|\tilde{d}^m, b^m, \lambda^{-1}I)dB
\]

\[
= \int \prod_{m=1}^{m_1} \mathcal{N}(b^m|\tilde{b}^m, \Sigma_B^m) \log \prod_{m=1}^{m_1} \left( (2\pi)^{-n/2}\lambda^{-1}I \right)^{-1/2} e^{-\frac{1}{2}(d^m - b^m)\Sigma_b^{-1}(d^m - b^m)} dB
\]

\[
= \int \prod_{m=1}^{m_1} \mathcal{N}(b^m|\tilde{b}^m, \Sigma_B^m) \left( -\frac{nm_1}{2} \log(2\pi\lambda^{-1}) - \frac{\lambda}{2} \sum_{m=1}^{m_1} (\tilde{d}^m - \tilde{b}^m)^\top (\tilde{d}^m - \tilde{b}^m) \right) dB
\]

\[
= -\frac{nm_1}{2} \log(2\pi\lambda^{-1}) - \frac{\lambda}{2} \sum_{m=1}^{m_1} \left( \text{Tr}(\Sigma_B^m) + (\tilde{d}^m - \tilde{b}^m)^\top (\tilde{d}^m - \tilde{b}^m) \right)
\]

where \( \Sigma_B^m = \Sigma_B + K_{XW}K_{WW}^{-1}G^mK_{WW}^{-1}K_{WX} \), \( \tilde{b}^m = K_{XW}K_{WW}^{-1}v^m \), and \( \tilde{d}^m = K_{XW}K_{WW}^{-1}v^m \), according to the variational distribution \( Q \).

We now express the variational lower bound of the log likelihood as follow:

\[
\mathcal{L} = \mathcal{L}_M + \mathcal{L}_G - \mathcal{L}_{KL} + \mathcal{L}_B - \frac{nm_1}{2} \log(2\pi\lambda^{-1}) \tag{18}
\]

where

\[
\mathcal{L}_M = \log p(M|K_ZK_{ZZ}^{-1}f) \tag{19}
\]

\[
\mathcal{L}_G = \log p(f = \tilde{f}|Z) = \log \mathcal{N}(f = \tilde{f}|0, K_{ZZ}) \tag{20}
\]

\[
= -\frac{1}{2} \tilde{f}^\top K_{ZZ}^{-1}\tilde{f} - \frac{n_{inducing}}{2} \log(2\pi) - \frac{1}{2} \log |K_{ZZ}| \tag{21}
\]

\[
\mathcal{L}_{KL} = KL(q(V)||p(V|W)) = \sum_{m=1}^{m_1} KL(\mathcal{N}(v^m|\tilde{v}^m, G^m)||\mathcal{N}(v^m|0, K_{WW})) \tag{22}
\]

\[
= \sum_{m=1}^{m_1} \frac{1}{2} \left( \text{Tr}(K_{WW}^{-1}(G^m + \tilde{v}^m\tilde{v}^m\top) - n_{inducing} + \log \left| \frac{K_{WW}}{|G^m|} \right| \right) \tag{23}
\]

\[
\mathcal{L}_B = -\frac{\lambda}{2} \sum_{m=1}^{m_1} \text{Tr}(\Sigma_B + K_{XW}K_{WW}^{-1}G^mK_{WW}^{-1}K_{WX}) \tag{24}
\]

which is also described in the main paper. The learning of the model involves optimizing over the variational parameters, including \( \tilde{f}, \tilde{v}^m, G^m \), inducing inputs \( Z \), as well as hyperparameters for the kernel functions, which is performed through backpropagation based on the gradients of the variational lower bound \( \mathcal{L} \) with respect to these parameters.

3 Optimizing the Variational Distribution \( q(V) \)

As can be seen, the variational lower bound \( \mathcal{L} \) depends on the parameters of the variational distribution \( q(V) = \prod_{m=1}^{m_1} \mathcal{N}(v^m|\tilde{v}^m, G^m) \), which can be optimized to improve the lower bound further. For the last term in \( \mathcal{L} \), we have...
\[
\int q(V) q(D) p(B|V, W, X) \log \frac{p(D|B)p(V|W)}{q(V)} \, d(D, B, V) \\
= \int q(V) \left( \int q(D) p(B|V, W, X) \log \frac{p(D|B)p(V|W)}{q(V)} \, d(D, B) \right) \, dV \\
= \int q(V) \left( \int p(B|V, W, X) \log \frac{p(D = \tilde{D}|B)p(V|W)}{q(V)} \, dB \right) \, dB \\
= \int q(V) e^{\langle \log p(D = \tilde{D}|B) \rangle_{p(B|V, W, X)} + \log p(V|W)} \, dV
\]

where we have \( \tilde{D} = [\tilde{d}^1 \ldots \tilde{d}^{m_1}] \), with \( \tilde{d}^m = K_{XW} K_{WW}^{-1} \tilde{e}^m \), and \( \tilde{e}^m \) for \( m = 1, \ldots, m_1 \) are variational parameters to optimize.

To maximize the above quantity, we can reverse the Jensen’s inequality to obtain the condition that:

\[
\log q(V) = \text{const} + \langle \log p(D = \tilde{D}|B) \rangle_{p(B|V, W, X)} + \log p(V|W)
\]

Now for the term \( \langle \log p(D = \tilde{D}|B) \rangle_{p(B|V, W, X)} \), we have:

\[
\langle \log p(D = \tilde{D}|B) \rangle_{p(B|V, W, X)} = \sum_{m=1}^{m_1} \langle \log N(\tilde{d}^m | b^m, \lambda^{-1} I) \rangle_{p(B|V, W, X)} \\
= \text{const} + \frac{1}{2} \sum_{m=1}^{m_1} \lambda^{-1} \text{Tr} \left( \tilde{d}^m \tilde{d}^m^\top + b^m b^{m\top} - 2 \tilde{d}^m b^{m\top} \right)_{N(b^m | K_{XW} K_{WW}^{-1} v^m, \Sigma_b)} \\
= \text{const} + \frac{1}{2} \sum_{m=1}^{m_1} \left( \lambda^{-1} \text{Tr} \left( \tilde{d}^m \tilde{d}^m^\top + \Sigma_B + v^m v^{m\top} K_{WW}^{-1} K_{XX} K_{WW}^{-1} v^m - 2 v^{m\top} K_{WW}^{-1} K_{WW} K_{WW}^{-1} v^m \right) \right)
\]

Therefore, we have:

\[
\log q(V^m) = \text{const} - \frac{1}{2} \left( \lambda v^{m\top} K_{WW}^{-1} K_{WW} K_{WW}^{-1} v^m - 2 \lambda v^{m\top} K_{WW}^{-1} v^m \right)
\]

Therefore by completing the squares we have \( q(V^m) = N(v^m | \tilde{v}^m, \Sigma_{v^m}) \):

\[
\Sigma_{v^m} = (K_{WW}^{-1} + \lambda K_{WW} K_{WW} K_{WW}^{-1})^{-1} \\
= \lambda^{-1} K_{WW} (\lambda^{-1} K_{WW} + K_{XX} K_{XX}^{-1})^{-1} K_{WW} \\
\tilde{v}^m = \lambda \Sigma_{v^m}^{-1} K_{WW} K_{XX} \tilde{d}^m \\
= K_{WW} (\lambda^{-1} K_{WW} + K_{XX} K_{XX}^{-1})^{-1} K_{XX} \tilde{d}^m
\]

With the above optimized variational parameters for \( q(V^m) \), we first obtain:

\[
\int q(V^m) \langle \log p(d^m = \tilde{d}^m | b^m) \rangle_{p(b^m | v^m, W, X)} \, dv^m = \\
= -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log |K_{WW}| - \frac{1}{2} \text{Tr} (K_{WW}^{-1} (\Sigma_{v^m} + \tilde{v}^m \tilde{v}^{m\top}) - 2 \tilde{v}^{m\top} K_{WW}^{-1} K_{WW} \tilde{d}^m)
\]

Next, we calculate \( \int q(V^m) \log p(v^m | W) \, dv^m \):

\[
\int q(V^m) \log p(v^m | W) = -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log |K_{WW}| - \frac{1}{2} \text{Tr} (K_{WW}^{-1} (\Sigma_{v^m} + \tilde{v}^m \tilde{v}^{m\top}))
\]

Finally we have:

\[
H(q(V^m)) = q(V^m) \log \frac{1}{q(V^m)} = \frac{n}{2} \log(2\pi) + \frac{1}{2} \log |\Sigma_{v^m}| \tag{25}
\]
Summarizing, we have:

\[
\int q(V)q(D)p(B|V, W, X) \log \frac{p(D|B)p(V|W)}{q(V)} d(D, B, V)
\leq \sum_{m=1}^{m_1} \left[ -\frac{n}{2} \log(2\pi \lambda^{-1}) - \frac{1}{2} \log |K_{WW}| - \frac{1}{2} Tr(K^{-1}_{WW}(\Sigma_{V^m} + \tilde{v}_s^T \tilde{v}_s^m)) + \frac{1}{2} \log |\Sigma_{V^m}| \\
- \frac{\lambda}{2} Tr \left( \tilde{d}_m \tilde{d}_m^T + \Sigma_B + K_{WW}^{-1} K_{WX} K_{WW}^{-1} (\Sigma_{V^m} + \tilde{v}_s^T \tilde{v}_s^m) - 2 \tilde{v}_s^m \tilde{v}_s^m^T K_{WW}^{-1} K_{WX} \tilde{d}_m \right) \right]
\]

We now express the variational lower bound of the log likelihood as follow:

\[
L = L_M + L_G + L_{DBV}
\]

where

\[
L_M = \log p(\mathcal{M}|K_D Z K_Z Z, \tilde{f})
\]

\[
L_G = \log p(u = \tilde{u}|Z) = \log \mathcal{N}(u = \tilde{u}|0, K_Z Z)
\]

\[
= -\frac{1}{2} \tilde{u}^T K_{ZZ}^{-1} \tilde{u} - \frac{K}{2} \log(2\pi) - \frac{1}{2} \log |K_{ZZ}|
\]

\[
L_{DBV} = \sum_{m=1}^{m_1} \left[ -\frac{n}{2} \log(2\pi \lambda^{-1}) - \frac{1}{2} \log |K_{WW}| - \frac{1}{2} Tr(K^{-1}_{WW}(\Sigma_{V^m} + \tilde{v}_s^T \tilde{v}_s^m)) + \frac{1}{2} \log |\Sigma_{V^m}| \\
- \frac{\lambda}{2} Tr \left( \tilde{d}_m \tilde{d}_m^T + \Sigma_B + K_{WW}^{-1} K_{WX} K_{WW}^{-1} (\Sigma_{V^m} + \tilde{v}_s^T \tilde{v}_s^m) - 2 \tilde{v}_s^m \tilde{v}_s^m^T K_{WW}^{-1} K_{WX} \tilde{d}_m \right) \right]
\]

The parameters we need to learn in this case include the variational parameters \( \tilde{f} \), and \( \tilde{e}_m \) for \( m = 1, \ldots, m_1 \), inducing inputs \( Z \), as well as hyperparameters for kernel functions.

4 Parameters Learning by Derivatives

In this section, we will obtain the derivatives of the marginal log likelihood \( L \) in (26) with respect to the variational parameters \( \tilde{f} \), \( \tilde{e}_m \) and inducing inputs \( Z \). The derivative of the reinforcement learning term, \( p(\mathcal{M}|r) \) in (7), with respect to the reward vectors \( r \), is given by:

\[
\frac{\partial}{\partial r} \log p(\mathcal{M}|r) = \sum_i \sum_t \left( \frac{\partial}{\partial r} r_{s_{i,t}, a_{i,t}} - \frac{\partial}{\partial r} V^r_{s_{i,t}} + \sum_{a'} \gamma T_{s_{i,t}, a_{i,t}} V^r_{s_{i,t}} \right)
\]

The first term, \( \sum_i \sum_t \frac{\partial}{\partial r} r_{s_{i,t}, a_{i,t}} \), is simply a vector that counts the number of state-action pairs in the demonstrations \( \hat{\mu} \), whose entry corresponding to \( (s, a) \) is given by: \( \hat{\mu}_{s,a} = \sum_i \sum_t 1_{s_{i,t}=s \land a_{i,t}=a} \). The second term involves the derivative of the value function at state \( s \) with respect to rewards, as indicated in (10), equal to the expected visitation count of each state-action pair when starting from state \( s \) and following the optimal stochastic policy, i.e., \( \frac{\partial}{\partial r} V^r_s = E[\mu|s] \), where \( \mu \) is a vector with each entry \( \mu_{s,a} \) corresponding to the expected visitation count for \( (s, a) \). Therefore, (31) can be written as:

\[
\frac{\partial}{\partial r} \log p(\mathcal{M}|r) = \hat{\mu} - \sum_i \sum_t E[\mu|s_{i,t}] + \sum_i \sum_t \sum_{a'} \gamma T_{s_{i,t}, a_{i,t}} E[\mu|s_{i,t}]
\]

\[
= \hat{\mu} - \sum_s \hat{\nu}_s E[\mu|s]
\]

where \( \hat{\nu}_s = \sum_a \hat{\mu}_{s,a} - \sum_i \sum_t \gamma T_{s_{i,t}, a_{i,t}} \). The term \( \sum_s \hat{\nu}_s E[\mu|s] \) can be computed efficiently by a simple iterative algorithm described in (4), which we do not recount here. Note that the above derivation follows from (10).
For the variational parameters $\tilde{f}$, we need to consider only two terms that involve it, i.e., $L_M, L_G$:

\[
\frac{\partial L_M}{\partial \tilde{f}} = \frac{\partial r}{\partial \tilde{f}} \frac{\partial L_M}{\partial r} = K_{DZ} K_{zz}^{-1} \frac{\partial \log p(M|r)}{\partial r}
\]

\[
\frac{\partial L_G}{\partial \tilde{f}} = -K_{zz}^{-1} \tilde{f}
\]

where $r = K_{DZ} K_{zz}^{-1} \tilde{f}$ is the reward vector that we use for reinforcement learning.

For the variational parameters $\tilde{e}_m$, let $\tilde{D} = \begin{bmatrix} K_{XW} K_{WW}^{-1} \tilde{e}_1, \ldots, K_{XW} K_{WW}^{-1} \tilde{e}_m \end{bmatrix} \in \mathbb{R}^{n \times m_1}$, and $E = [\tilde{e}_1, \ldots, \tilde{e}_m] \in \mathbb{R}^{K \times m_1}$:

\[
\frac{\partial L_M}{\partial E} = \frac{\partial \tilde{D}}{\partial E} \frac{\partial K_{DZ}}{\partial r} \frac{\partial r}{\partial K_{DZ}} \frac{\partial L_M}{\partial r}
\]

In addition, by applying matrix derivatives,

\[
\frac{\partial L_{DBV}}{\partial e^m} = \frac{\lambda}{2} \left( 2 K_{WW}^{-1} K_{XW} K_{WW}^{-1} + 2 K_{WW}^{-1} K_{XW} K_{WW}^{-1} \Gamma K_{XW} K_{WW}^{-1} \Gamma K_{XW} K_{WW}^{-1} \right) e^m - K_{WW}^{-1} K_{XW} K_{WW}^{-1} \Gamma K_{WW} K_{XW} K_{WW}^{-1} \Gamma K_{XW} K_{WW}^{-1} e^m
\]

The gradients are provided to minFunc [3], which calls a quasi-Newton strategy, where limited-memory BFGS updates with Shanno-Phua scaling are used in computing the step direction, and a bracketing line-search for a point satisfying the strong Wolfe conditions is used to compute the step direction.

References


