What should be transferred in transfer learning?

Chris Williams and Kian Ming A. Chai

July 2009

Motivation

- Is learning the N-th thing any easier than learning the first? (Thrun, 1996)
- Gain strength by sharing information across tasks
- Examples of multi-task learning
  - Co-occurrence of ores (geostats)
  - Object recognition for multiple object classes
  - Personalization (personalizing spam filters, speaker adaptation in speech recognition)
  - Compiler optimization of many computer programs
  - Robot inverse dynamics (multiple loads)
- Are task descriptors available?

Outline

- Co-kriging
- Intrinsic Correlation Model
- Multi-task learning:
  - 1. MTL as Hierarchical Modelling
  - 2. MTL as Input-space Transformation
  - 3. MTL as Shared Feature Extraction
- Multi-task learning in Robot Inverse Dynamics

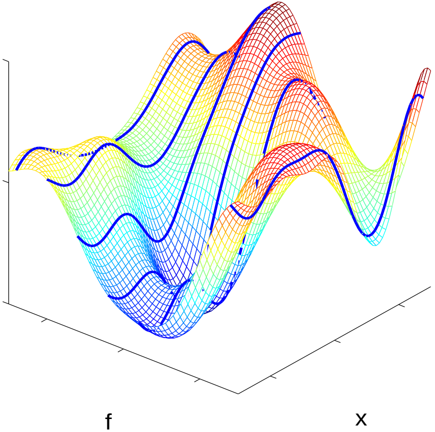
Co-kriging

Consider M tasks, and N distinct inputs $\mathbf{x}_1, \ldots, \mathbf{x}_N$:

- $f_{i\ell}$ is the response for the ℓ^{th} task on the i^{th} input $\mathbf{x}_i$
- Gaussian process with covariance function

$$k(\mathbf{x}, \ell; \mathbf{x}', m) = \langle f_\ell(\mathbf{x}) f_m(\mathbf{x}') \rangle$$

- Goal: Given noisy observations $\mathbf{y}$ of $\mathbf{f}$ make predictions of unobserved values $\mathbf{f}_*$ at locations X_*
- Solution Use the usual GP prediction equations


f
x

Covariance functions and hyperparameters

- The squared-exponential covariance function

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp[-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^T M (\mathbf{x} - \mathbf{x}')]$$

  is often used in machine learning
- Many other choices, e.g. Matern family, rational quadratic, non-stationary cov fns etc
- if M is diagonal, the entries are inverse squared lengthscales $\rightarrow$ *automatic relevance determination* (ARD, Neal 1996)
- Estimation of *hyperparameters* by optimization of log marginal likelihood

$$L = -\frac{1}{2}\mathbf{y}^T K_y^{-1}\mathbf{y} - \frac{1}{2}\log|K_y| - \frac{n}{2}\log 2\pi$$

Some questions

- What kinds of (cross)-covariance structures match different ideas of multi-task learning?
- Are there multi-task relationships that don't fit well with co-kriging?

Intrinsic Correlation Model (ICM)

$$\langle f_\ell(\mathbf{x}) f_m(\mathbf{x}')\rangle = K^f_{\ell m} k^x(\mathbf{x},\mathbf{x}') \qquad y_{i\ell} \sim \mathcal{N}(f_\ell(\mathbf{x}_i), \sigma^2_\ell),$$

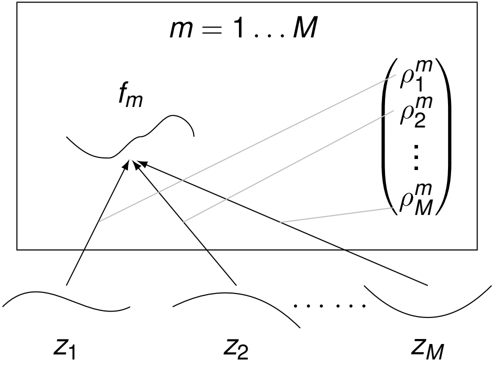
- K^f: PSD matrix that specifies the inter-task similarities (could depend parametrically on task descriptors if these are available)
- k^x: Covariance function over inputs
- σ^2_ℓ: Noise variance for the ℓ^{th} task.
- Linear Model of Coregionalization is a sum of ICMs

ICM as a linear combination of indepenent GPs

- Independent GP priors over the functions $z_j(\mathbf{x}) \Rightarrow$ multi-task GP prior over $f_m(\mathbf{x})$s

$$\langle f_\ell(\mathbf{x}) f_m(\mathbf{x}') \rangle = K^f_{\ell m} k^x(\mathbf{x}, \mathbf{x}')$$

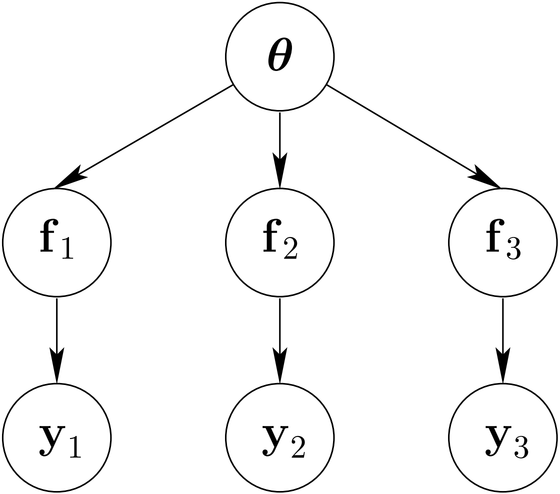
- $K^f \in \mathbb{R}^{M \times M}$ is a task (or context) similarity matrix with $K^f_{\ell m} = (\boldsymbol{\rho}^m)^T \boldsymbol{\rho}^\ell$



- Some problems conform nicely to the ICM setup, e.g. robot inverse dynamics (Chai, Williams, Klanke, Vijayakumar 2009; see later)
- Semiparametric latent factor model (SLFM) of Teh et al (2005) has P latent processes each with its own covariance function. Noiseless outputs are obtained by linear mixing of these latent functions

1. Multi-task Learning as Hierarchical Modelling

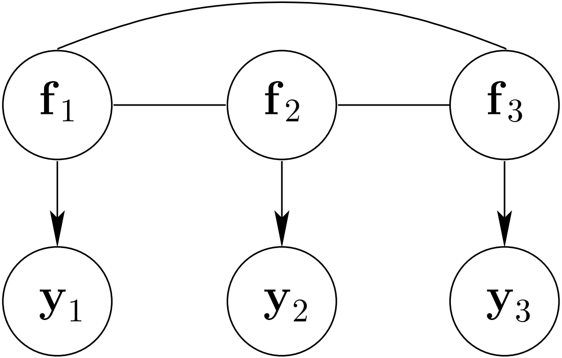
e.g. Baxter (JAIR, 2000), Evgeniou et al (JMLR, 2005), Goldstein (2003)



- Prior on θ may be generic (e.g. isotropic Gaussian) or more structured
- Mixture model on $\theta \rightarrow$ task clustering
- Task clustering can be implemented in the ICM model using a block diagonal K^f, where each block is a cluster
- Manifold model for θ, e.g. linear subspace $\Rightarrow$ low-rank structure of K^f (e.g. linear regression with correlated priors)
- Combination of the above ideas $\rightarrow$ a mixture of linear subspaces
- If task descriptors are available then can have $K^f_{\ell m} = k^f(\mathbf{t}_\ell, \mathbf{t}_m)$

GP view

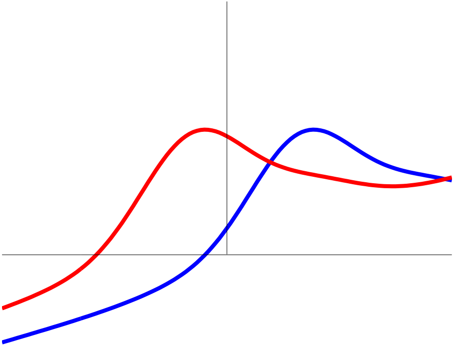
Integrate out θ



2. MTL as Input-space Transformation

- Ben-David and Schuller (COLT, 2003), $f_2(\mathbf{x})$ is related to $f_1(\mathbf{x})$ by a $\mathcal{X}$-space transformation $f : \mathcal{X} \rightarrow \mathcal{X}$
- Suppose $f_2(\mathbf{x})$ is related to $f_1(\mathbf{x})$ by a *shift* $\mathbf{a}$ in $\mathbf{x}$-space
- Then

$$\langle f_1(\mathbf{x}) f_2(\mathbf{x}') \rangle = \langle f_1(\mathbf{x}) f_1(\mathbf{x}' - \mathbf{a}) \rangle = k_1(\mathbf{x}, \mathbf{x}' - \mathbf{a})$$



- More generally can consider *convolutions*, e.g.

$$f_i(\mathbf{x}) = \int h_i(\mathbf{x} - \mathbf{x}')g(\mathbf{x}')d\mathbf{x}'$$

  to generate dependent f's (e.g. Ver Hoef and Barry, 1998; Higdon, 2002; Boyle and Frean, 2005). $\delta(\mathbf{x} - \mathbf{a})$ is a special case

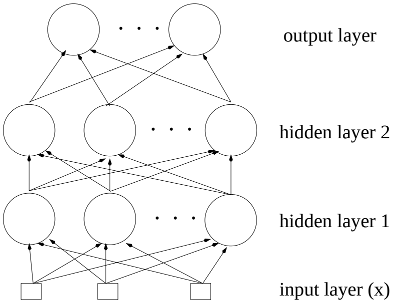
- Alvarez and Lawrence (2009) generalize this to allow a linear combination of several latent processes

$$f_i(\mathbf{x}) = \sum_{r=1}^{R} \int h_{ir}(\mathbf{x} - \mathbf{x}')g_r(\mathbf{x}')d\mathbf{x}'$$

- ICM and SPFM are special cases using the $\delta()$ kernel

3. Shared Feature Extraction

- Intuition: multiple tasks can depend on the same extracted features; all tasks can be used to help learn these features
- If data is scarce for each task this should help learn the features
- Bakker and Heskes (2003) – neural network setup



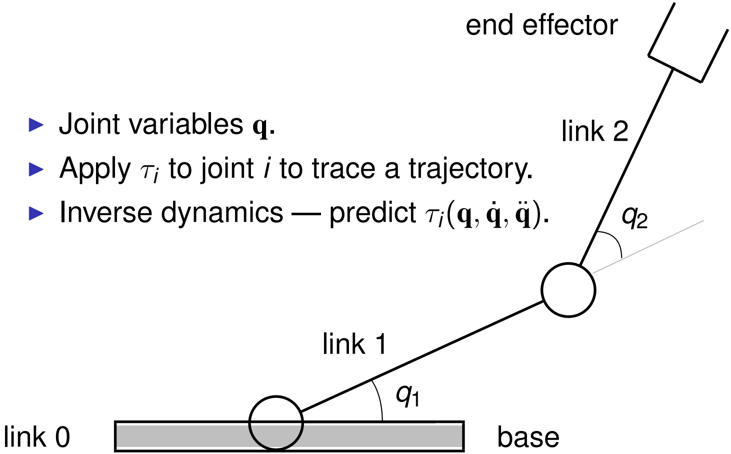
- Minka and Picard (1999): assume that the multiple tasks are independent GPs but with *shared* hyperparameters
- Yu, Tresp and Schawaighofer (2005) extend this so that all tasks share the same kernel hyperparameter, but can have different kernels
- Could also have inter-task correlations
- Interesting case if different tasks have different **x**-spaces; convert from each task-dependent **x**-space to same feature space?

Discussion

- 3 types of multi-task learning setup
- ICM and convolutional cross-covariance functions, shared feature extraction
- Are there multi-task relationships that don't fit well with a co-kriging framework?

Multi-task Learning in Robot Inverse Dynamics

- Joint variables $\mathbf{q}$.
- Apply τ_i to joint i to trace a trajectory.
- Inverse dynamics — predict $\tau_i(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$.



- Torques are non-linear functions of $\mathbf{x} \stackrel{\text{def}}{=} (\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$.
- (One) idealized rigid body control:

$$\tau_i(\mathbf{x}) = \underbrace{\mathbf{b}_i^{\mathrm{T}}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{q}}^{\mathrm{T}} H_i(\mathbf{q})\dot{\mathbf{q}}}_{\text{kinetic}} + \overbrace{g_i(\mathbf{q})}^{\text{potential}} + \underbrace{f_i^{\mathsf{v}}\dot{q}_i + f_i^{\mathsf{c}}\mathrm{sgn}(\dot{q}_i)}_{\text{viscous and Coulomb frictions}},$$

- Physics-based modelling can be hard due to factors like unknown parameters, friction and contact forces, joint elasticity, making analytical predictions unfeasible
- This is particularly true for compliant, lightweight humanoid robots

- Functions *change* with the loads handled at the end effector
- Loads have different mass, shapes, sizes.
- Bad news (1): Need a different inverse dynamics model for different loads.
- Bad news (2): Different loads may go through different trajectory in data collection phase and may explore different portions of the $\mathbf{x}$-space.

- Good news: the changes enter through changes in the dynamic parameters of the last link
- Good news: changes are linear wrt the dynamic parameters

$$\tau_i^m(\mathbf{x}) = \mathbf{y}_i^T(\mathbf{x})\boldsymbol{\pi}^m$$

  where $\boldsymbol{\pi}^m \in \mathbb{R}^{11}$ (e.g. Petkos and Vijayakumar,2007)
- Reparameterization:

$$\tau_i^m(\mathbf{x}) = \mathbf{y}_i^T(\mathbf{x})\boldsymbol{\pi}^m = \mathbf{y}_i^T(\mathbf{x})A_i^{-1}A_i\boldsymbol{\pi}^m = \mathbf{z}_i^T(\mathbf{x})\boldsymbol{\rho}_i^m$$

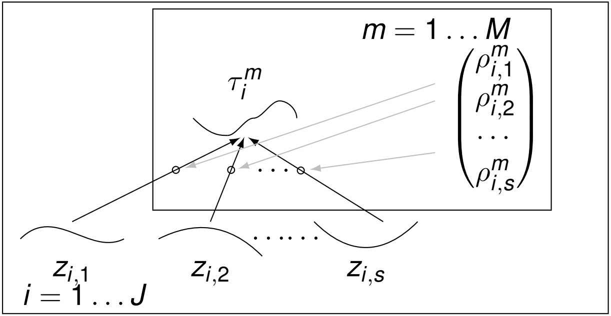
  where A_i is a non-singular 11×11 matrix

GP prior for Inverse Dynamics for multiple loads

- Independent GP priors over the functions $z_{ij}(\mathbf{x}) \Rightarrow$ multi-task GP prior over τ_i^ms

$$\left\langle \tau_i^\ell(\mathbf{x})\tau_i^m(\mathbf{x}') \right\rangle = (K_i^\rho)_{\ell m} k_i^x(\mathbf{x}, \mathbf{x}')$$

- $K_i^\rho \in \mathbb{R}^{M \times M}$ is a task (or context) similarity matrix with $(K_i^\rho)_{\ell m} = (\boldsymbol{\rho}_i^m)^T \boldsymbol{\rho}_i^\ell$



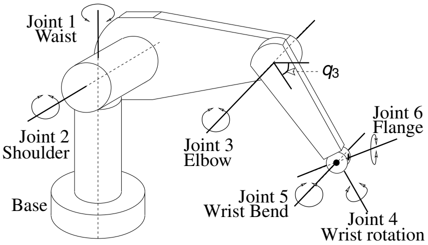
GP prior for $k(\mathbf{x}, \mathbf{x}')$

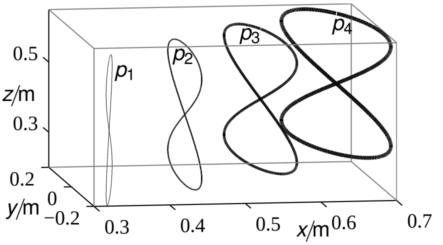
$$
\begin{aligned}
k(\mathbf{x}, \mathbf{x}') = \text{bias} &+ [\text{linear with ARD}](\mathbf{x}, \mathbf{x}') \\
&+ [\text{squared exponential with ARD}](\mathbf{x}, \mathbf{x}') \\
&+ [\text{linear (with ARD)}](\mathrm{sgn}(\dot{q}), \mathrm{sgn}(\dot{q}'))
\end{aligned}
$$

- Domain knowledge relates to last term (Coulomb friction)

Data

- Puma 560 robot arm manipulator: 6 degrees of freedom
- Realistic simulator (Corke, 1996), including viscous and asymmetric-Coulomb frictions.
- 4 paths $\times$ 4 speeds $=$ 16 different trajectories:
- Speeds: 5s, 10s, 15s and 20s completion times.
- 15 loads (contexts): 0.2kg . . . 3.0kg, various shapes and sizes.





Data

Training data

- 1 reference trajectory common to handling of all loads.
- 14 unique training trajectories, one for each context (load)
- 1 trajectory has no data for any context; thus this is always novel

Test data

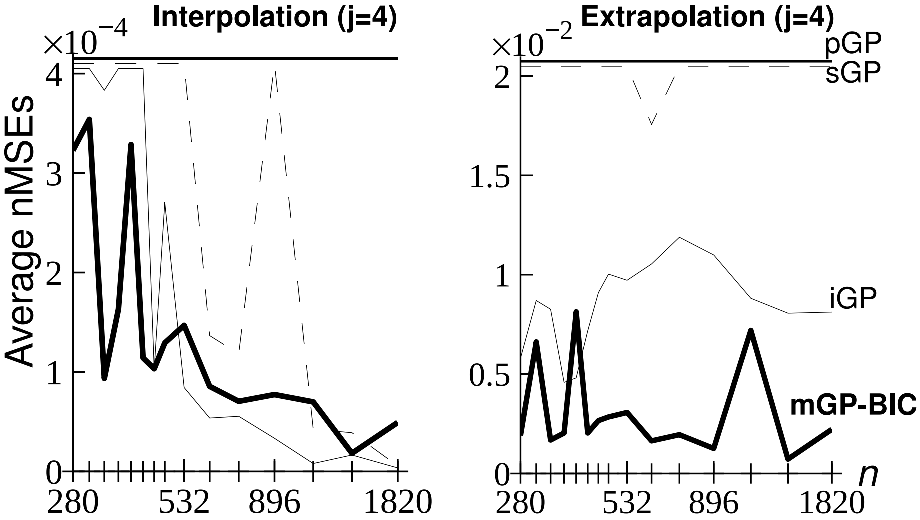
- Interpolation data sets for testing on reference trajectory and the unique trajectory for each load.
- Extrapolation data sets for testing on all trajectories.

Methods

sGP	Single task GPs	GPs trained separately for each load
iGP	Independent GP	GPs trained independently for each load but tying parameters across loads
pGP	pooled GP	one single GP trained by pooling data across loads
mGP	multi-task GP with BIC	sharing latent functions across loads, selecting similarity matrix using BIC

- For mGP, the rank of K^f is determined using BIC criterion

Results



Interpolation (j=4)
×10^−4
Average nMSEs
4
3
2
1
0
280
532
896
1820
Extrapolation (j=4)
×10^−2
pGP
sGP
2
1.5
1
0.5
0
iGP
mGP-BIC
n
280
532
896
1820

Conclusions and Discussion

- GP formulation of MTL with factorization $k^x(\mathbf{x}, \mathbf{x}')$ and K^f, and encoding of task similarity
- This model fits exactly for multi-context inverse dynamics
- Results show that MTL can be effective
- This is one model for MTL, but what about others, e.g. cov functions that don't factorize?