GP positioning system for Cellular Networks (Schwaighofer et al)

Setting: Network of base stations (e.g. WLAN, DECT).

Goal: Predict spatial position of mobile station using measured signal strength.

Calibration: $D_j = \{y_i, \mathbf{x}_i\}_{i=1}^N$ measured signal strengths y_i at positions $\mathbf{x}_i$ from base station j.

Model: $y_i = s(x_i) + e_i$ with Gaussian noise and GP kernel $E[s(\mathbf{x})s(\mathbf{x}')] = K(\mathbf{x}, \mathbf{x}')$. Note: A **mean** $m_j(\mathbf{x})$ has to be added!

Matern Kernel: $K(z) = \frac{2(\sqrt{\nu}z)^\nu}{\Gamma(\nu)} K_\nu(2\sqrt{\nu}z)$ with $z = \sqrt{\lambda}||\mathbf{x} - \mathbf{x}'||$.

Posterior for signal strength at position $\mathbf{x} \to p_j(s_j(\mathbf{x})|D_j)$ (Gaussian). Prediction of most likely position:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \prod_j p_j(s_j(\mathbf{x})|D_j)$$

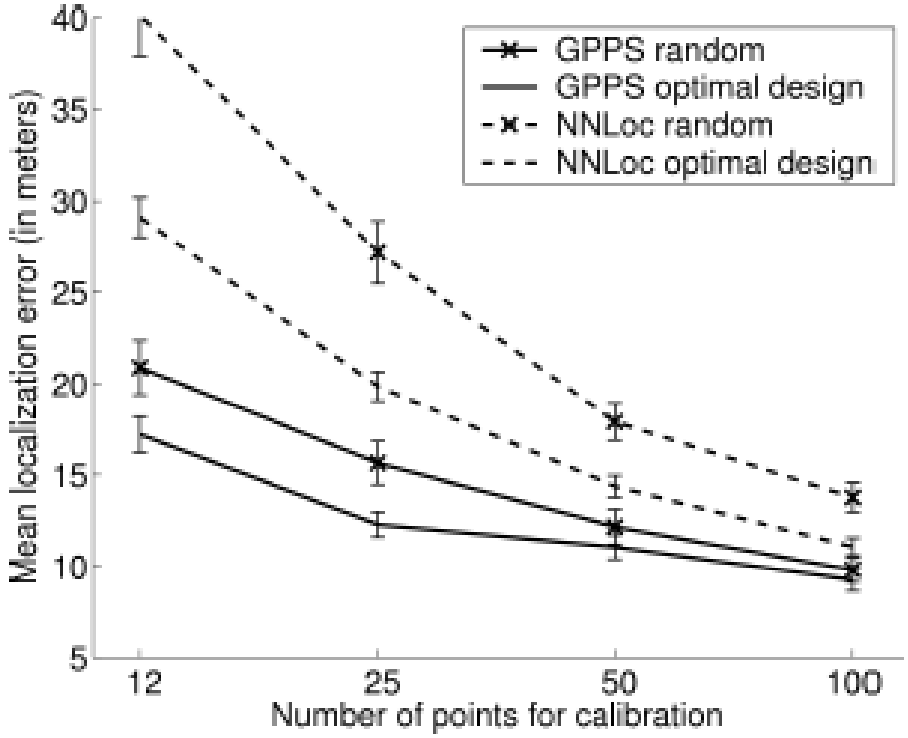
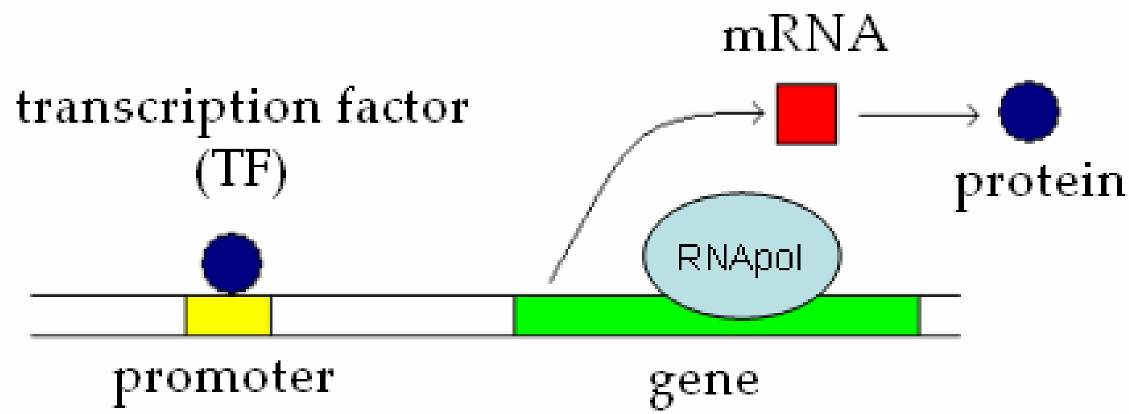


Figure 1: Mean localization error of the GPPS and the NNLoc method, as a function of the number of calibration points used. Vertical bars indicate ± 1 standard deviation of the mean localization error. The calibration points are either selected at random, or according to an optimal design criterion

Inference of transcriptional regulation using Gaussian processes

(Lawrence, Sanguinetti & Rattray)



- Transcription factors regulate genes by binding to specific sites.

- Hard to measure transcription factor activity directly. Inference must be based on measurement of mRNA concentration of target genes at discrete times $y_{ik} = x_i(t_k) +$ noise.

- Model equation (ordinary differential equation)

$$\frac{dx_i}{dt} = B_i - D_i x_i(t) + S_i f(t)$$

where $f(t)$ is the transcription factor activity.

- Model $f(t)$ by a Gaussian process. Since the differential equation is linear, $x(t)$ and $f(t)$ are jointly Gaussian processes.

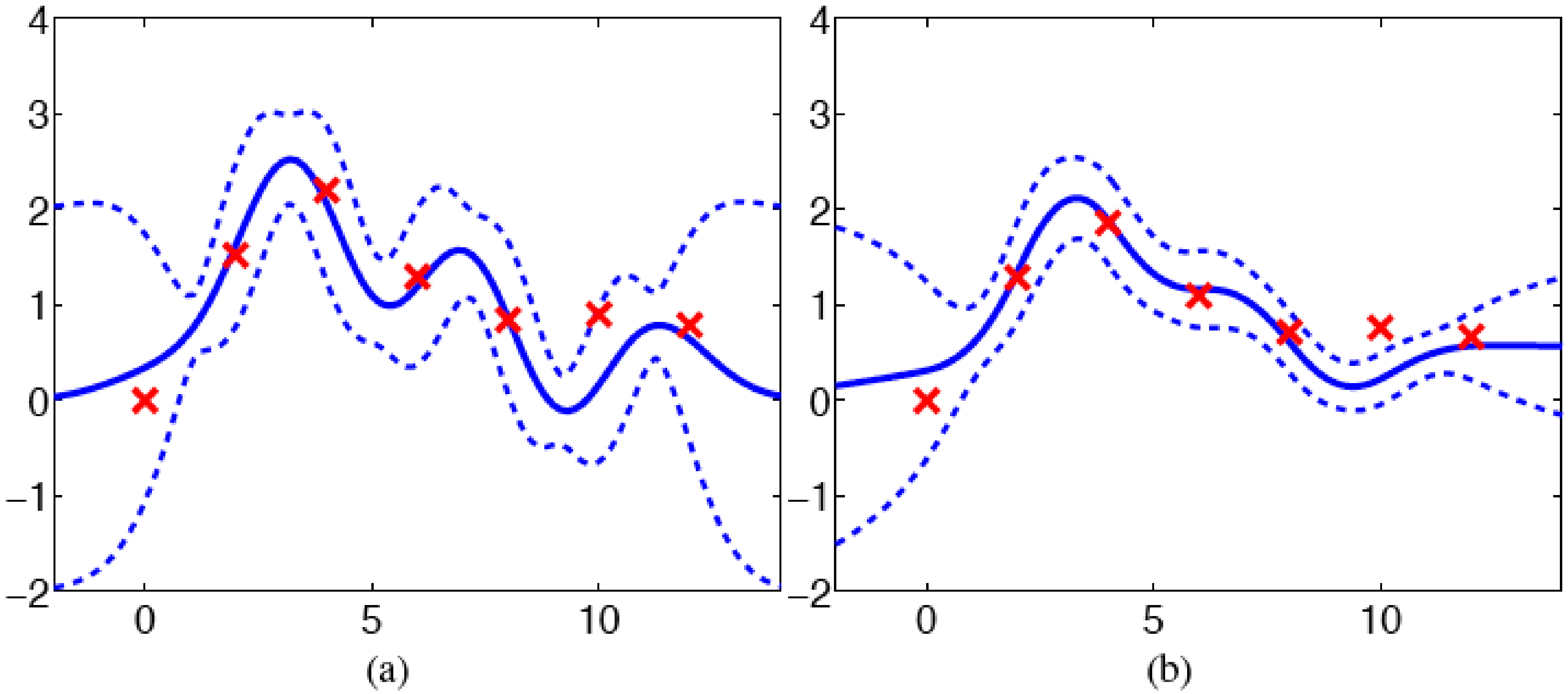


Figure 1: Predicted protein concentration for p53 using a linear response model: (a) squared exponential prior on f; (b) MLP prior on f. Solid line is mean prediction, dashed lines are 95% credibility intervals. The prediction of Barenco *et al.* was pointwise and is shown as crosses.

GP Emulators

O'Hagan & Kennedy (see e.g.
`http://www.tonyohagan.co.uk/academic/GEM/index.html`

and the MUCM (MANAGING UNCERTAINTY IN COMPLEX MODELS) page
`http://mucm.group.shef.ac.uk/`

Emulate complex simulation software packages. These evaluate functions $y = f(x)$ using very lengthy computations.

Learn a Gaussian process approximation $y = m(x) + \mathcal{GP}(0, K)$ from a small set of data.

Sensitivity analysis: Changes of outputs under small input changes.

Uncertainty analysis: Uncertainty of outputs based on uncertainty in inputs modelled by distribution $p(x)$.

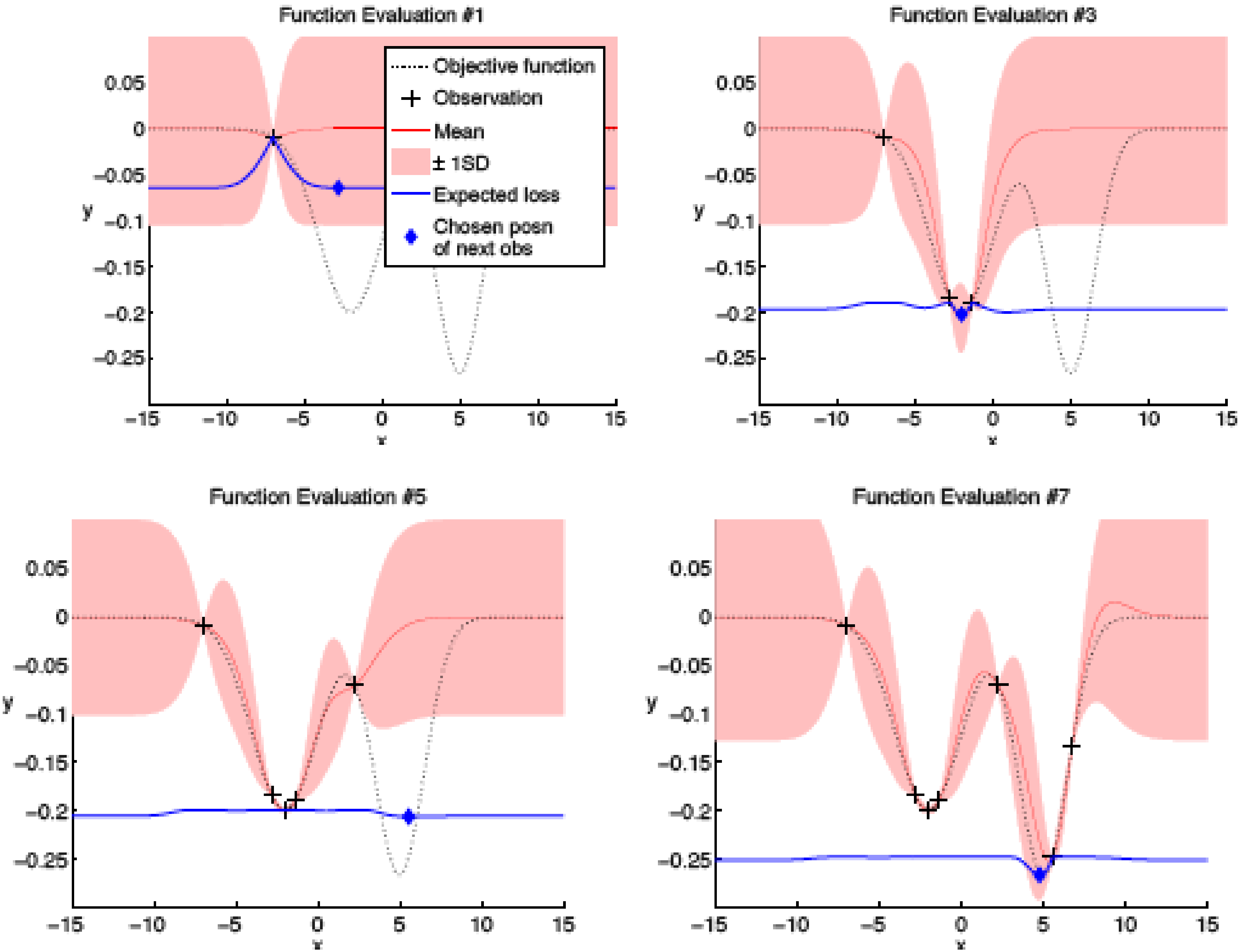
Gaussian Processes for Global Optimisation

Gaussian process (GP) models: Flexible Bayesian machine learning approach. Allows for estimating functions from data. Also provides confidence intervals.

- Problem: Find global optimum when function evaluations are costly.

- (Osborne et al:) Use function evaluations to approximate unknown function $f(x)$ by a GP $y(x)$.

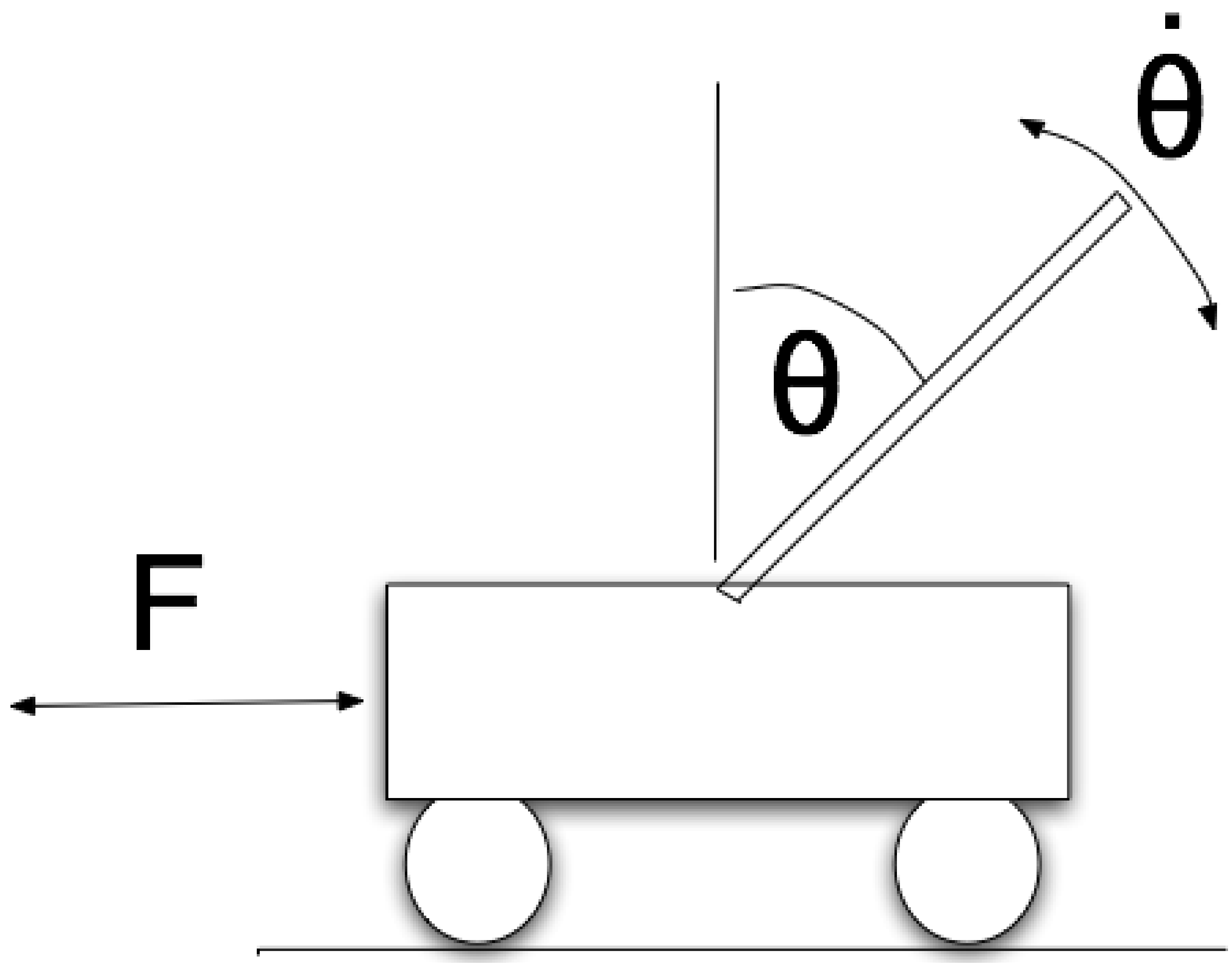
- Find new candidate point x_{n+1} for minimiser by minimising posterior expectation of *risk* $= \min\{y(x), f(x_n)\}$ with respect to x. This will take both mean and uncertainty of $y(x)$ into account.



Function Evaluation #1
Function Evaluation #3
Function Evaluation #5
Function Evaluation #7
Objective function
Observation
Mean
± 1SD
Expected loss
Chosen posn of next obs
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.25
y
−15
−10
−5
0
5
10
15
x

Solving control problems

(Learning to balance a pole, by Marc Deisenroth et al)



'Standard' approach

Let $\mathbf{x}_k = (\theta, \dot{\theta})$

- Use the exact ODE to get $\mathbf{x}_{k+1} = f\left(\mathbf{x}_k, \text{Force}(\mathbf{x}_k)\right)$

- Use dynamic programming to find Force($\mathbf{x}_k$) which minimises expected costs.

  Bellman equation:

$$V_k^*(\mathbf{x}_k) = \min_u E\left[g(\mathbf{x}_k, \mathbf{u}_k) + V_{k+1}^*(f(\mathbf{x}_k, \mathbf{u}_k))\right]$$

- **Problem:** Needs exact knowledge of dynamics (ODEs). Exact solution of Bellman equation computationally hard. It requires (continuous state space).

Gaussian process approach

1. Create **example** time series with random force.

2. **Emulator:** Train a Gaussian process (GP) regression model on examples to learn the dynamics $\mathbf{x}_{k+1} = f(\mathbf{x}_k, \text{Force}(\mathbf{x}_k))$ (with uncertainty).

3. Use GP to interpolate/extrapolate $Q(\mathbf{x}, \mathbf{u}) \doteq g(\mathbf{x}, \mathbf{u}) + V^*_{k+1}(f(\mathbf{x}, \mathbf{u}))$ from discrete set of $\mathbf{x}$ and $\mathbf{u}$. The uncertainty in the GP can be used to compute the expectation. in the Bellman equation.