

Gaussian Process Change Point Models

Yunus Saatçi



Haifa, Israel

June 24, 2010

Joint work with Ryan Turner and Carl Edward Rasmussen

Introduction & Motivation

- Many standard time series models are stationary by default.
- Nonstationarity is often encountered in many real-world time series.
- Nonstationarity defined as changes in generative parameters.
- Inability to react to such changes can hurt predictive performance.
- Examples: finance (CDOs, LTCM, . . .), robotics, networks etc.
- Many of these applications require an [online](#) response to nonstationarity.

- A time series model usually offers $p(x_{t+\delta}|x_{t-w}, \dots, x_t, \theta_m)$, δ and $w \geq 1$.

- A time series model usually offers $p(x_{t+\delta}|x_{t-w}, \dots, x_t, \theta_m)$, δ and $w \geq 1$.
- Combine such predictive capability with a **sequential** change point model.

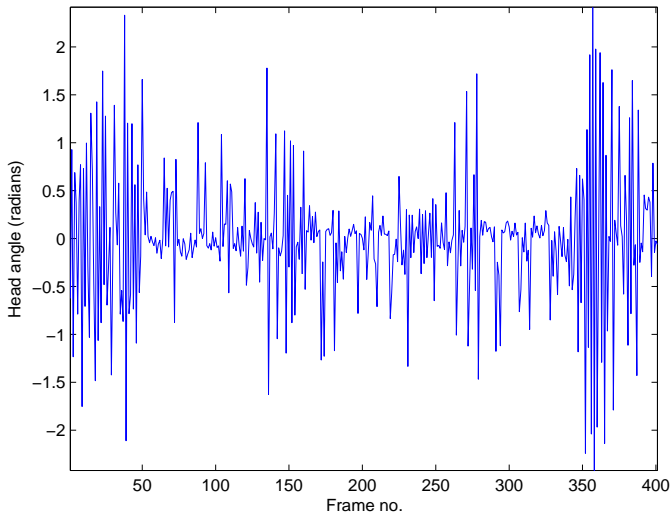
- A time series model usually offers $p(x_{t+\delta}|x_{t-w}, \dots, x_t, \theta_m)$, δ and $w \geq 1$.
- Combine such predictive capability with a **sequential** change point model.
- Resulting predictions will be robust to underlying regime changes.

- A time series model usually offers $p(x_{t+\delta}|x_{t-w}, \dots, x_t, \theta_m)$, δ and $w \geq 1$.
- Combine such predictive capability with a **sequential** change point model.
- Resulting predictions will be robust to underlying regime changes.
- Change point model used will be an extension of Adams & MacKay 2007.
- Important to combine uncertainty in parameter changes with uncertainty over their values.

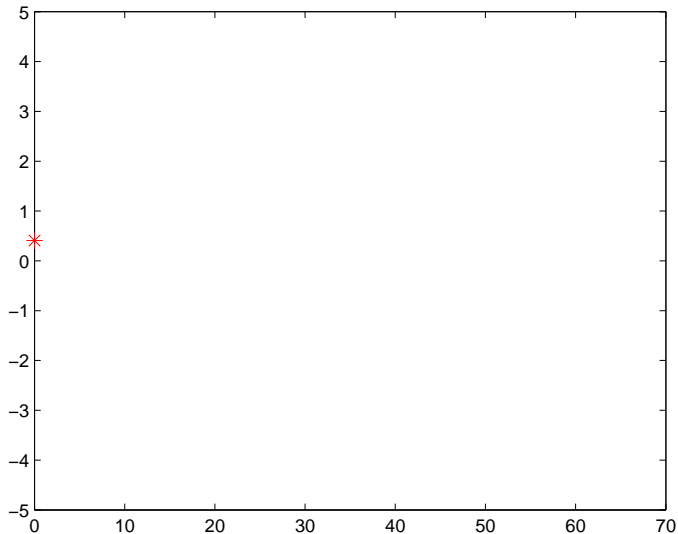
- A time series model usually offers $p(x_{t+\delta}|x_{t-w}, \dots, x_t, \theta_m)$, δ and $w \geq 1$.
- Combine such predictive capability with a **sequential** change point model.
- Resulting predictions will be robust to underlying regime changes.
- Change point model used will be an extension of Adams & MacKay 2007.
- Important to combine uncertainty in parameter changes with uncertainty over their values.
- Thus will use **Bayesian** perspective on time series.

- A time series model usually offers $p(x_{t+\delta}|x_{t-w}, \dots, x_t, \theta_m)$, δ and $w \geq 1$.
- Combine such predictive capability with a **sequential** change point model.
- Resulting predictions will be robust to underlying regime changes.
- Change point model used will be an extension of Adams & MacKay 2007.
- Important to combine uncertainty in parameter changes with uncertainty over their values.
- Thus will use **Bayesian** perspective on time series.
- Many time series models (e.g., $AR(p)$, $ARMA(p,q)$) are Gaussian processes (GPs).
- Provides desired Bayesian perspective.

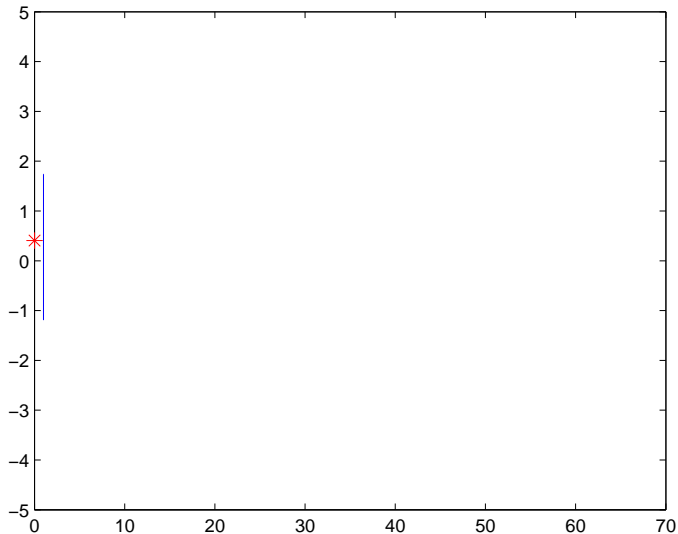
Motivating Example – Bee Waggle Dance



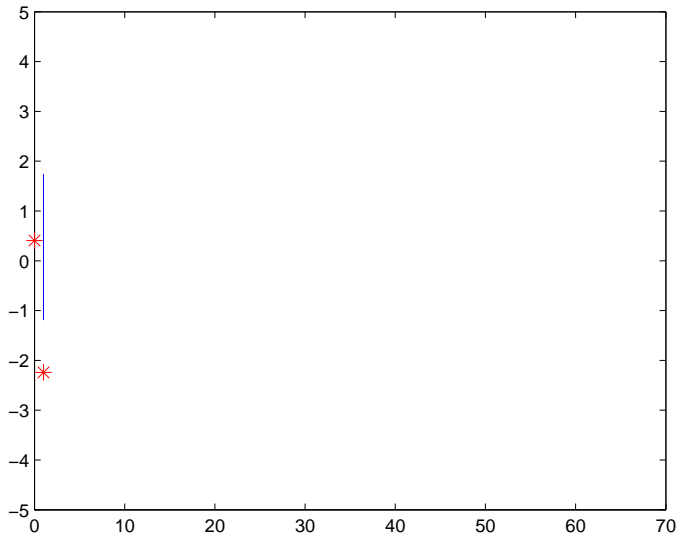
Motivating Example



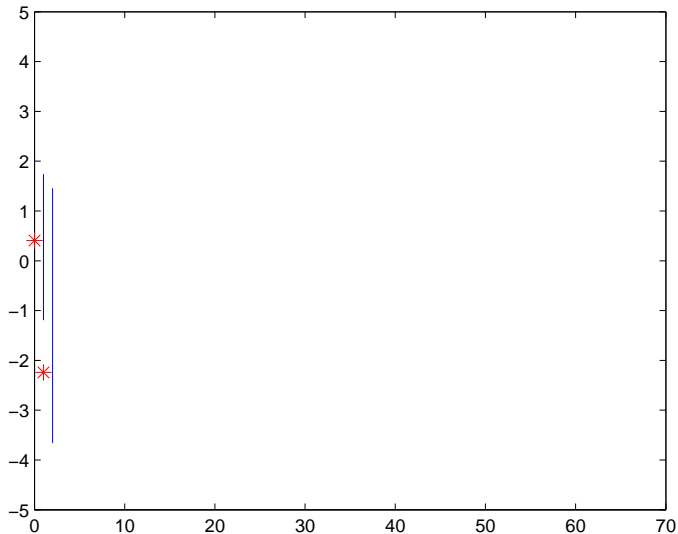
Motivating Example



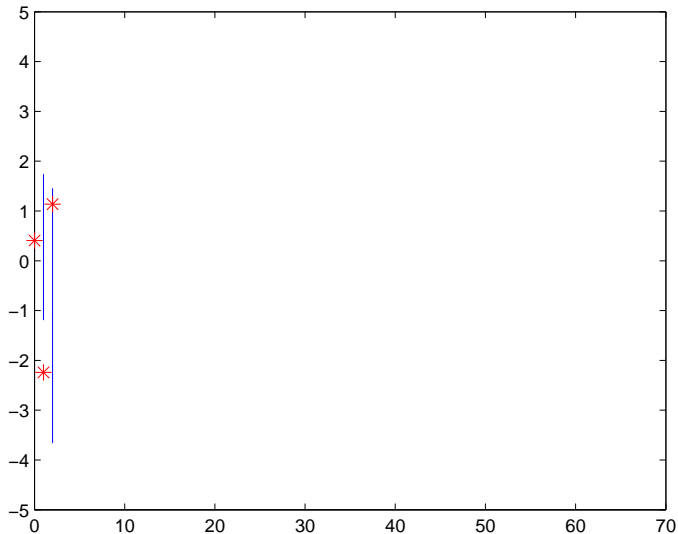
Motivating Example



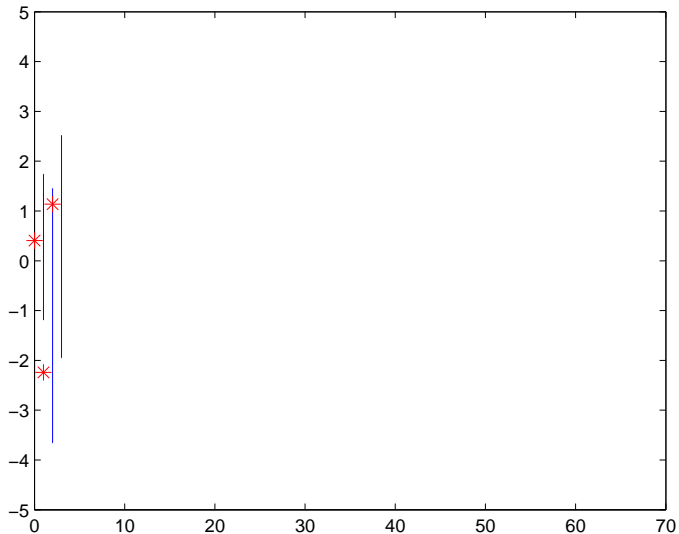
Motivating Example



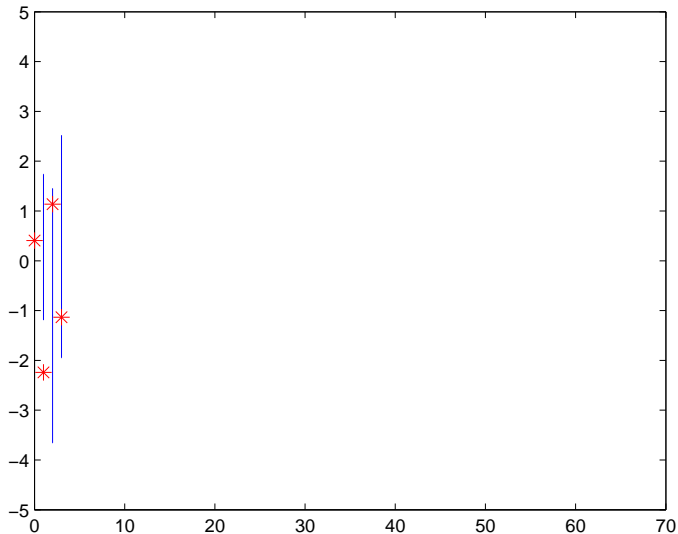
Motivating Example



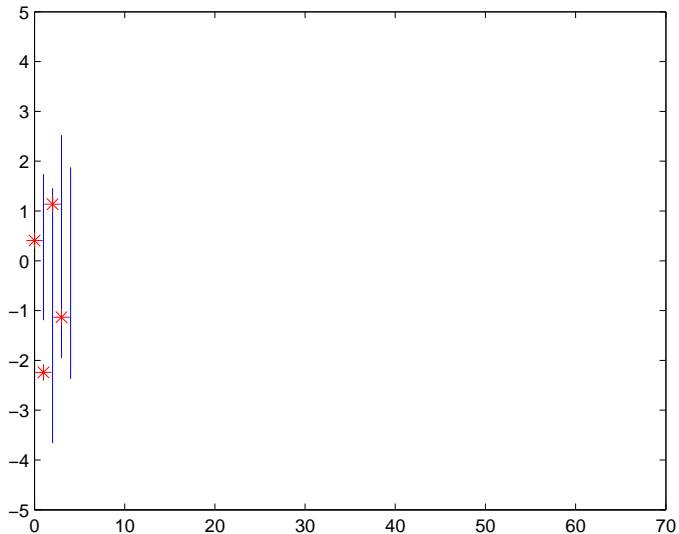
Motivating Example



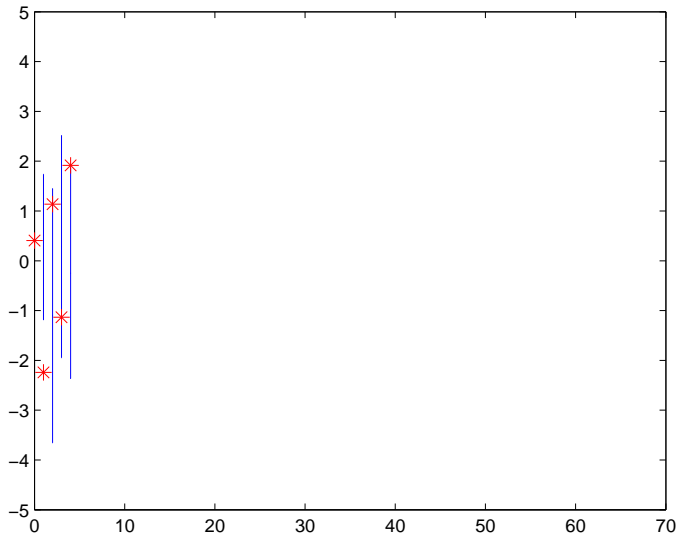
Motivating Example



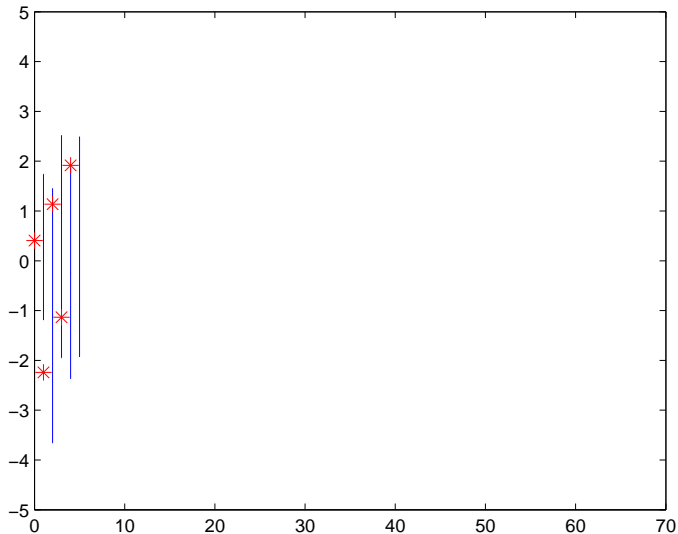
Motivating Example



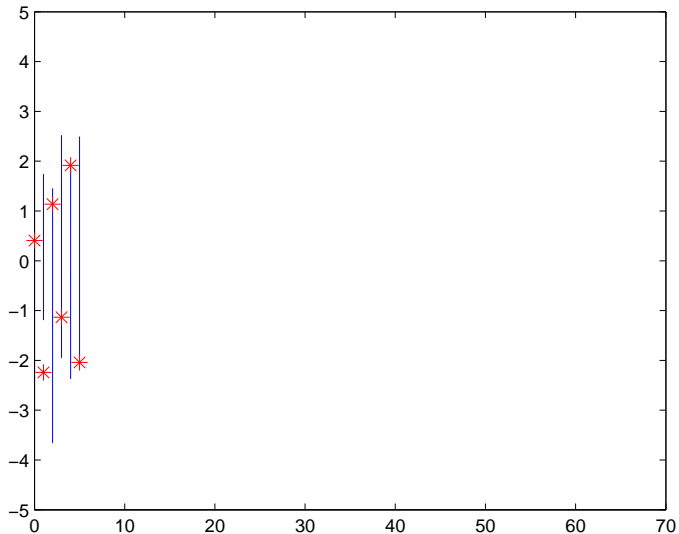
Motivating Example



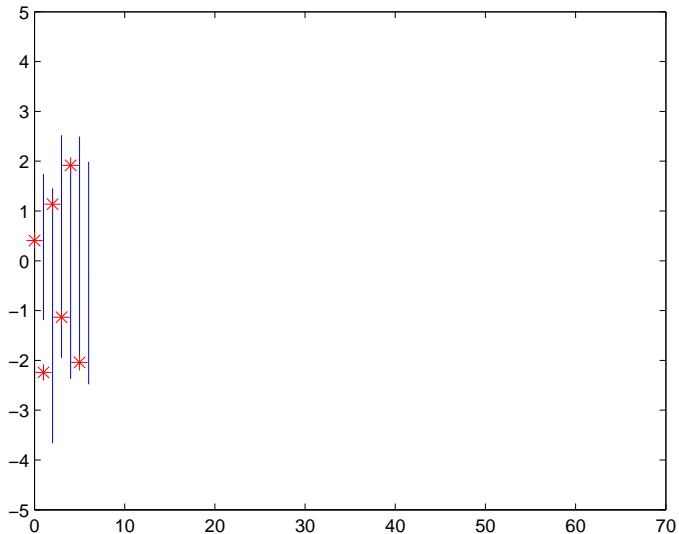
Motivating Example



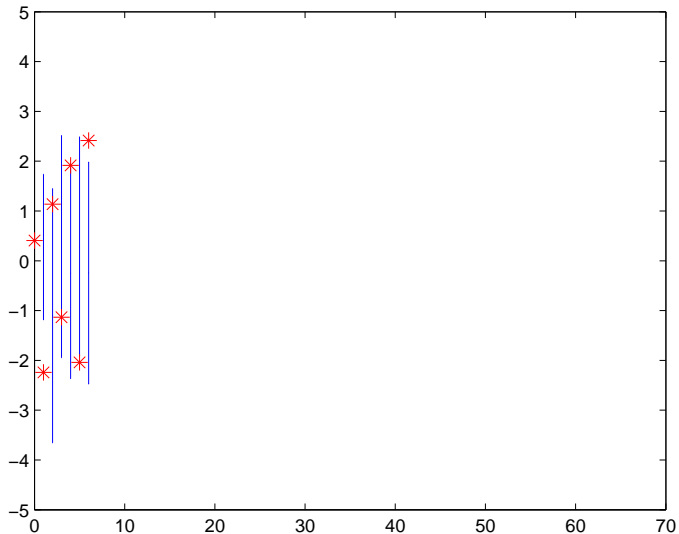
Motivating Example



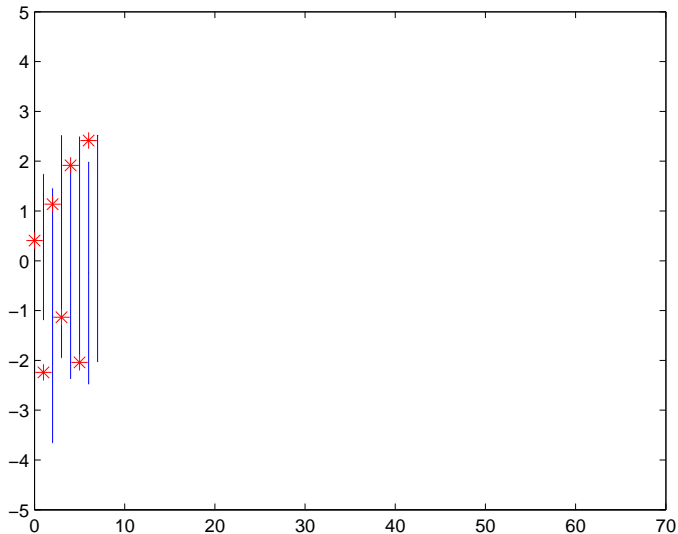
Motivating Example



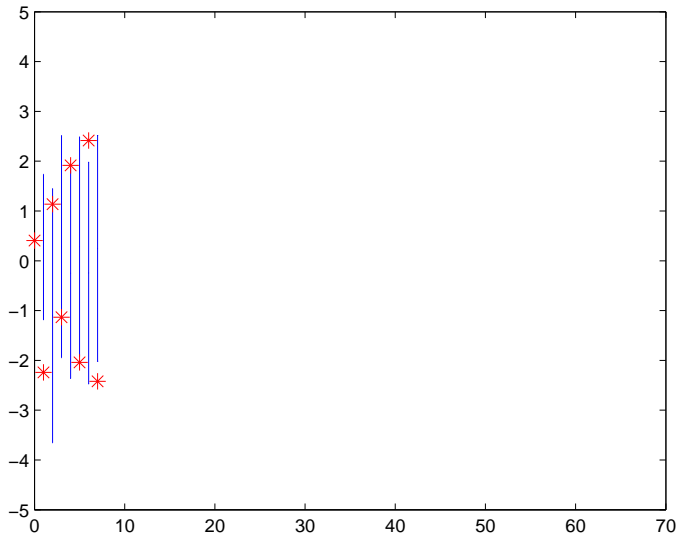
Motivating Example



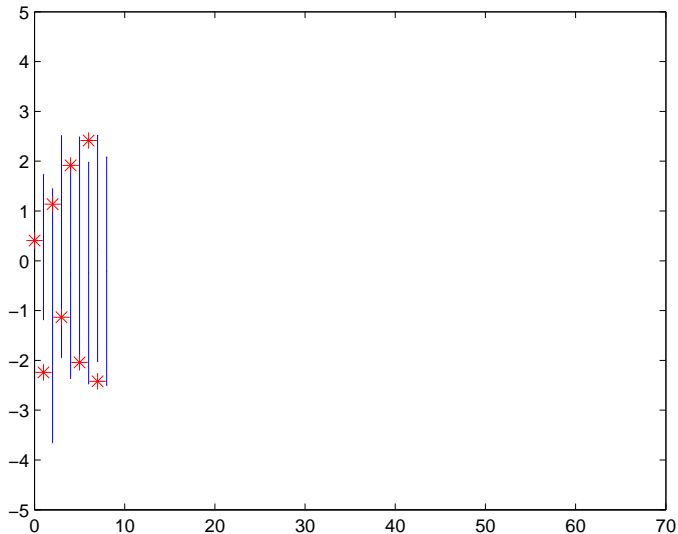
Motivating Example



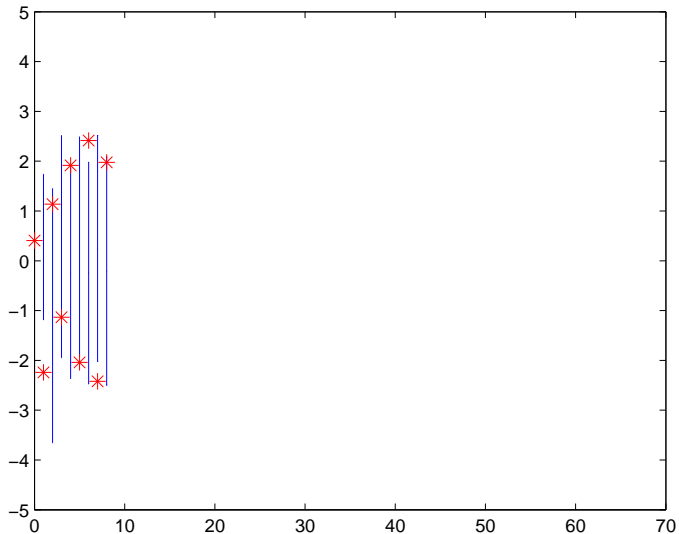
Motivating Example



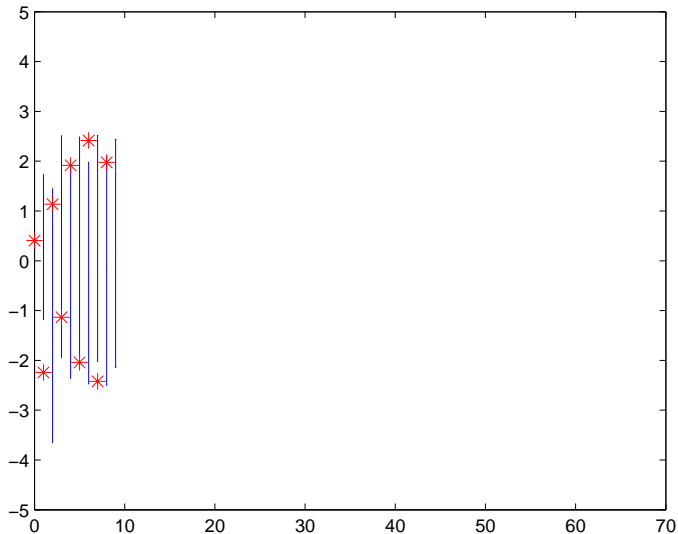
Motivating Example



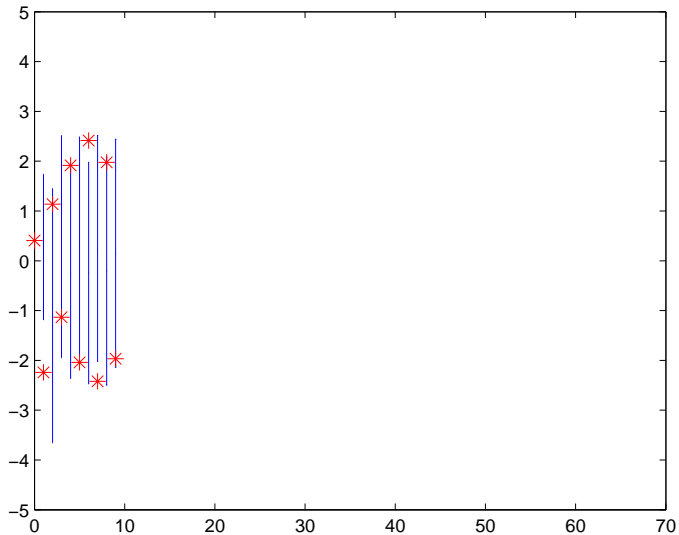
Motivating Example



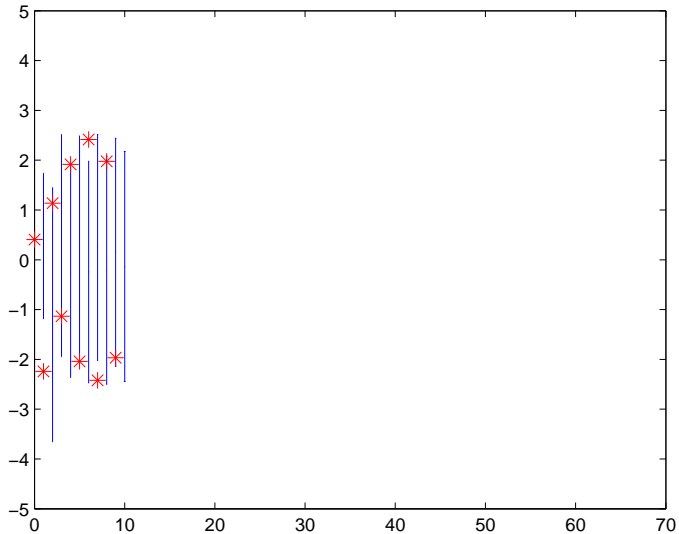
Motivating Example



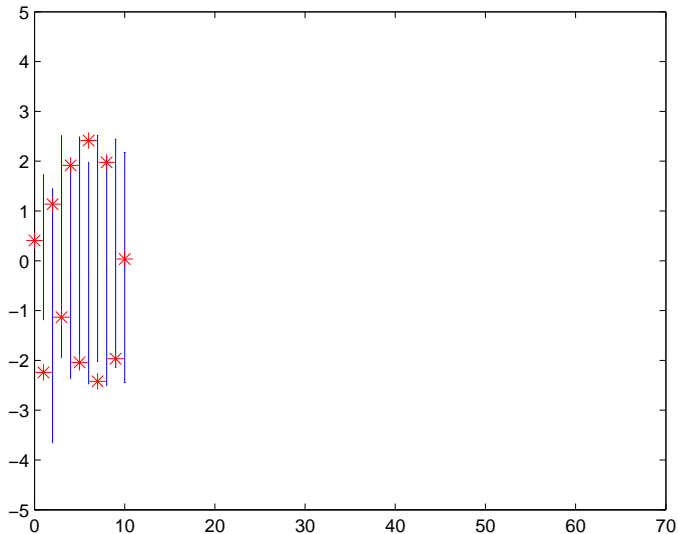
Motivating Example



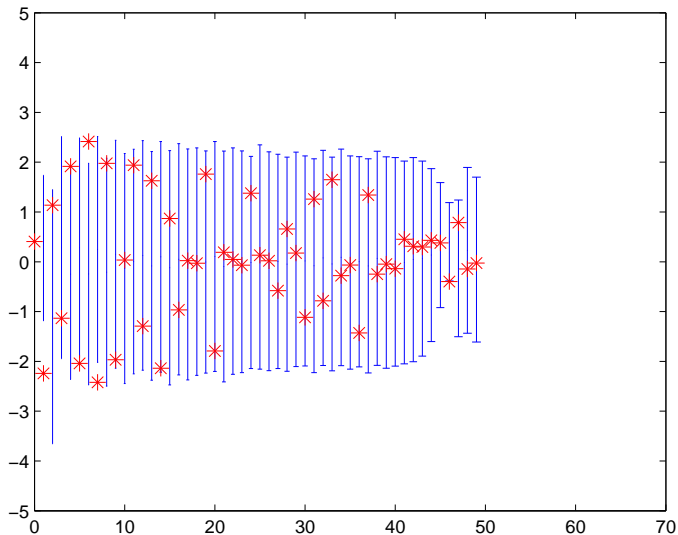
Motivating Example



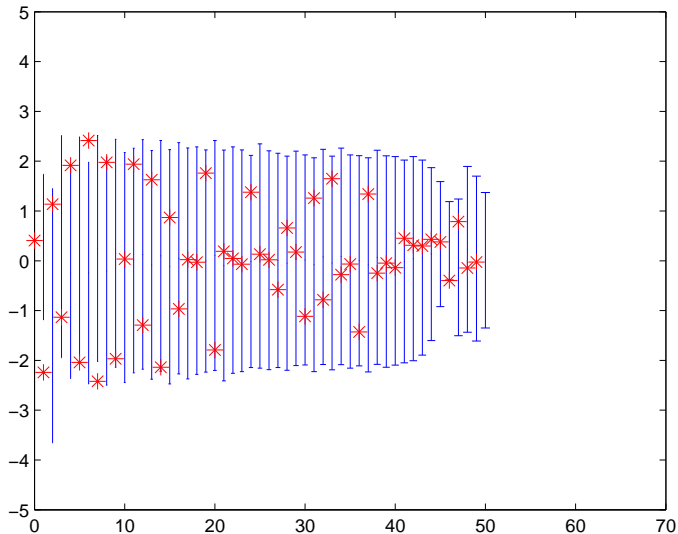
Motivating Example



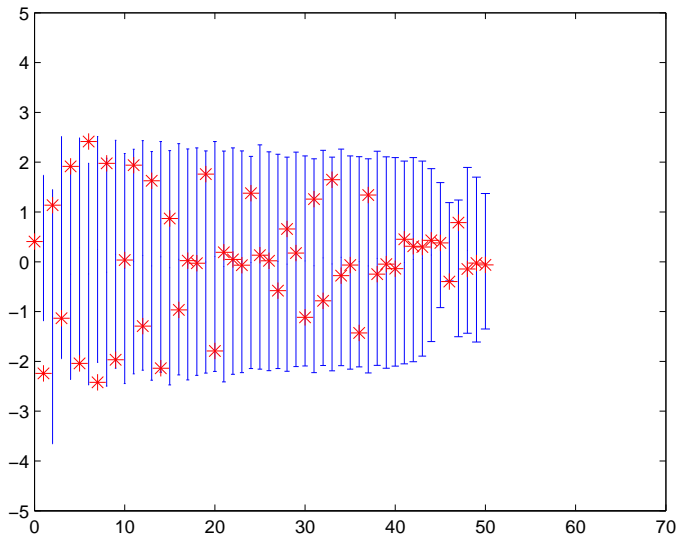
Motivating Example



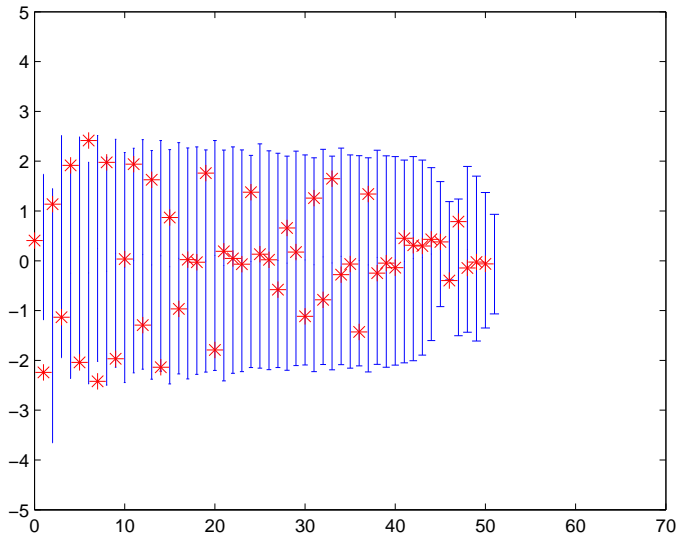
Motivating Example



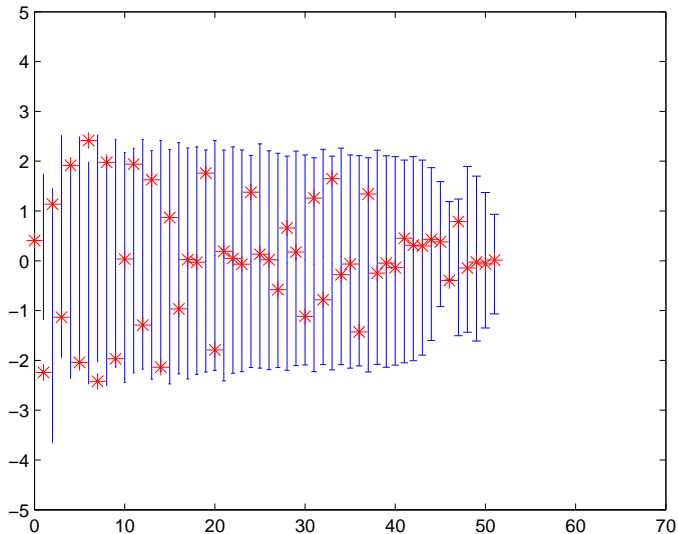
Motivating Example



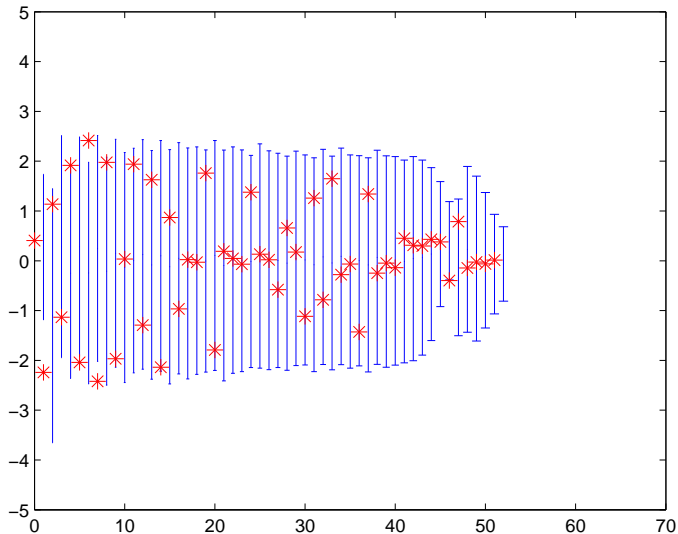
Motivating Example



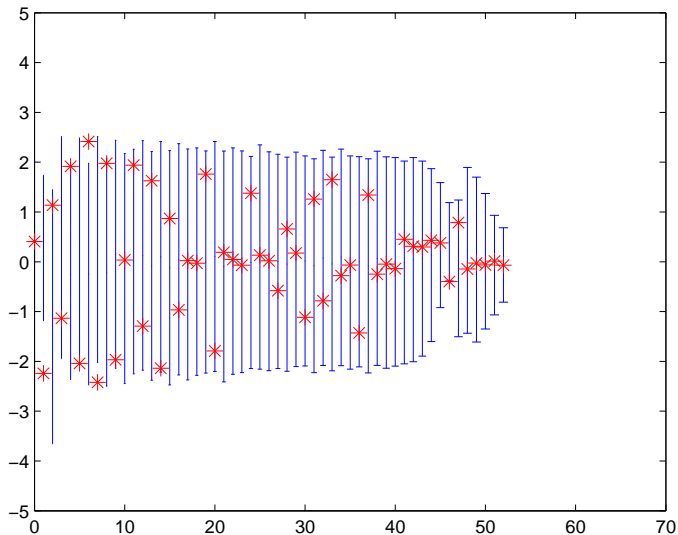
Motivating Example



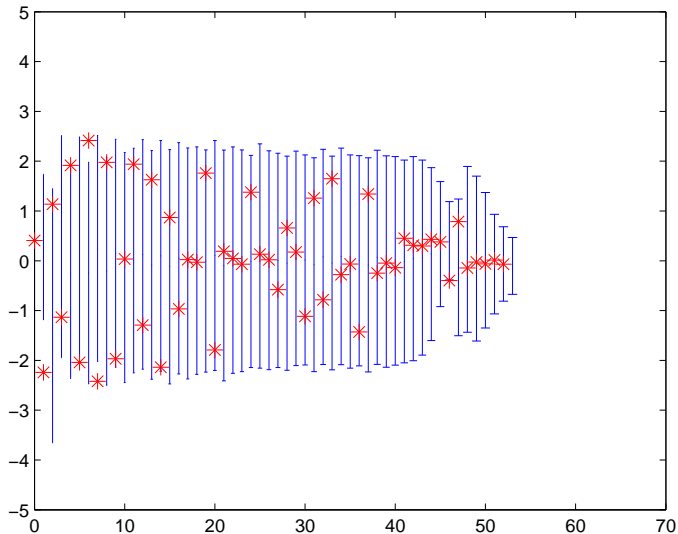
Motivating Example



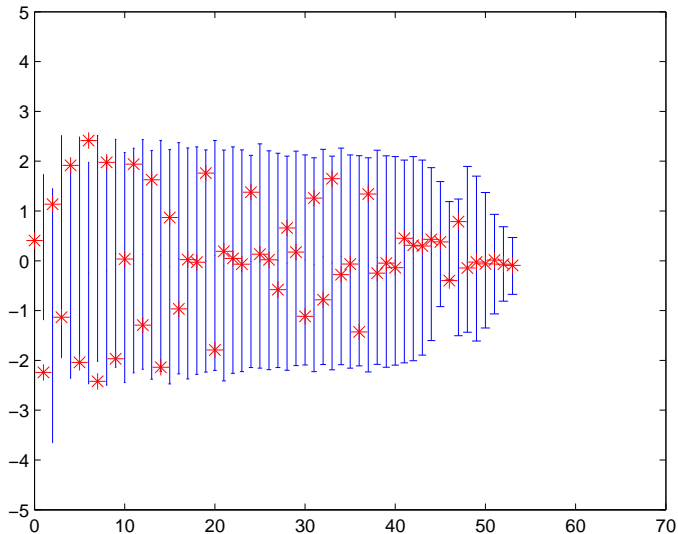
Motivating Example



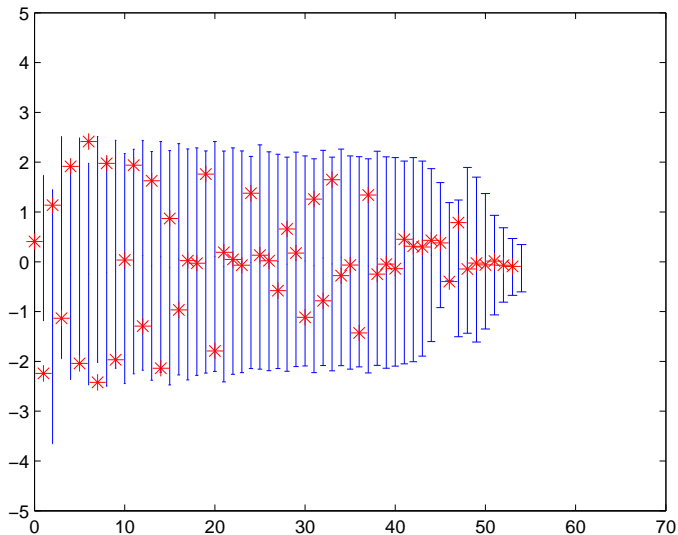
Motivating Example



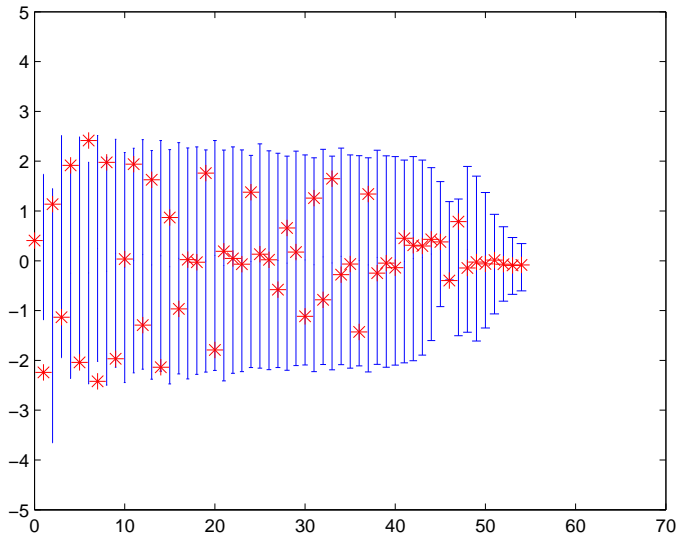
Motivating Example



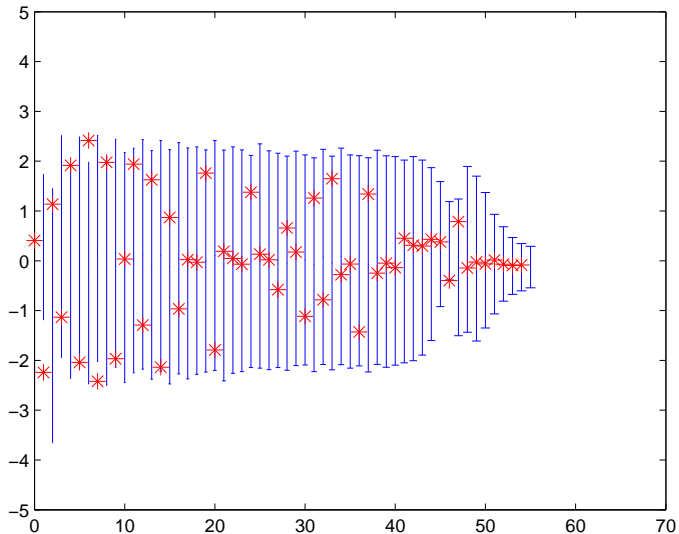
Motivating Example



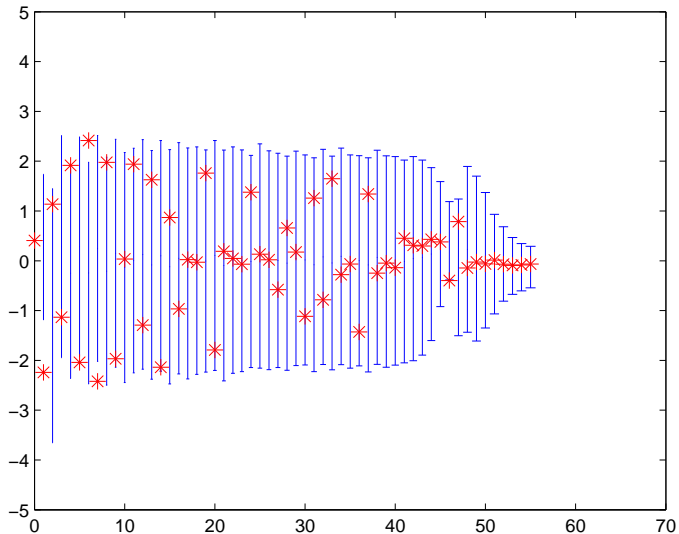
Motivating Example



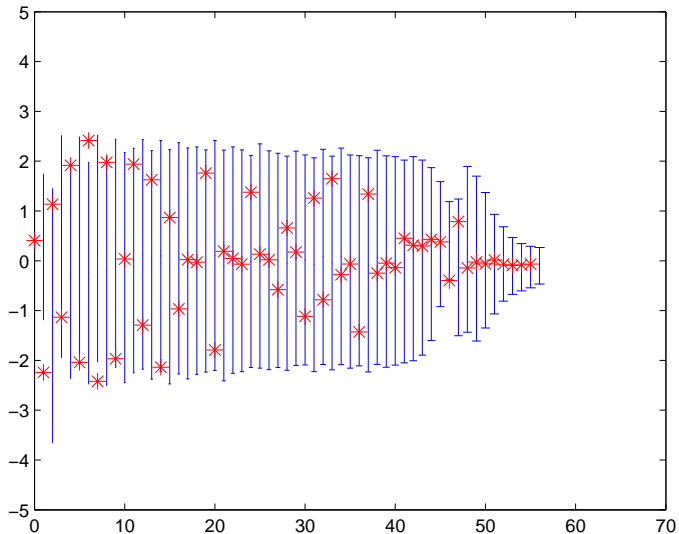
Motivating Example



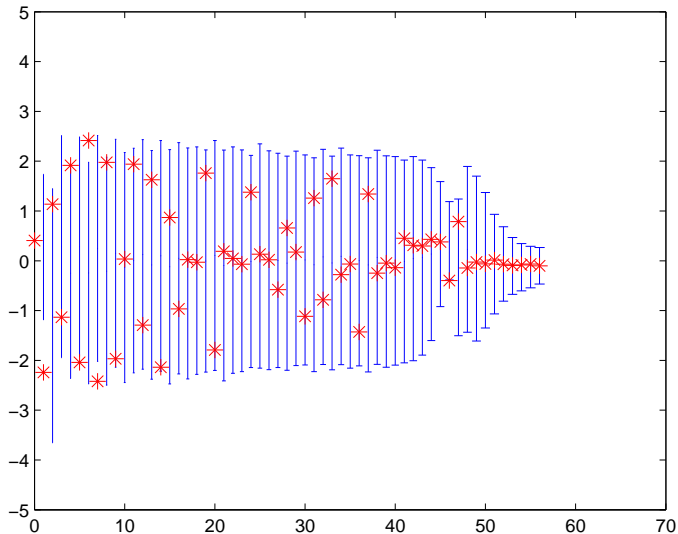
Motivating Example



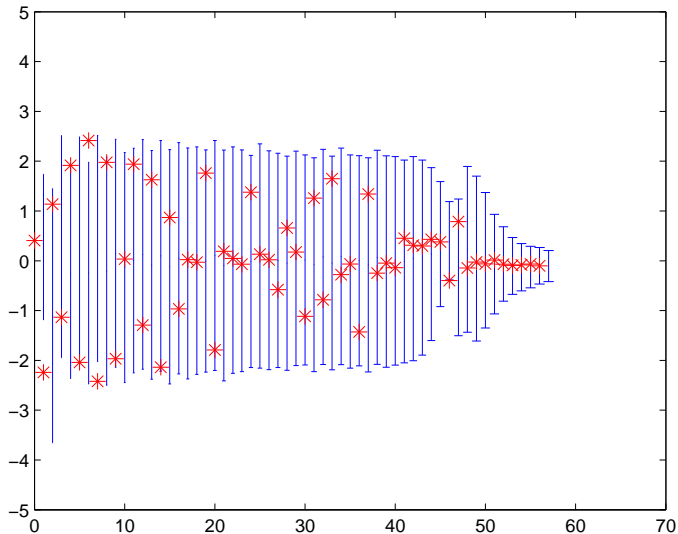
Motivating Example



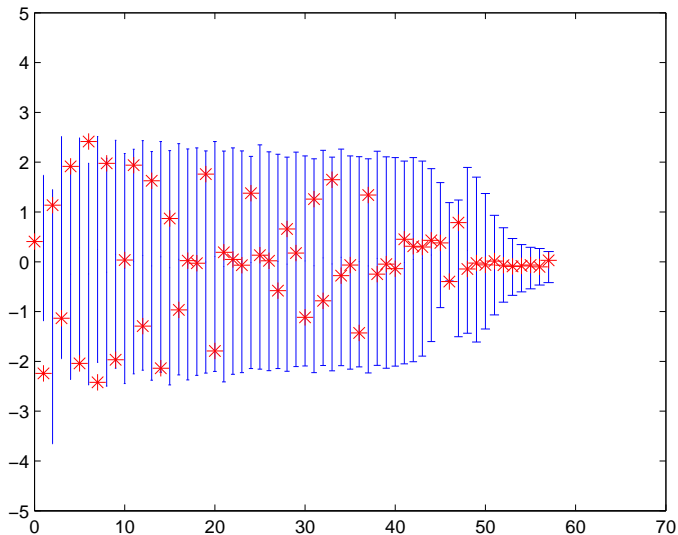
Motivating Example



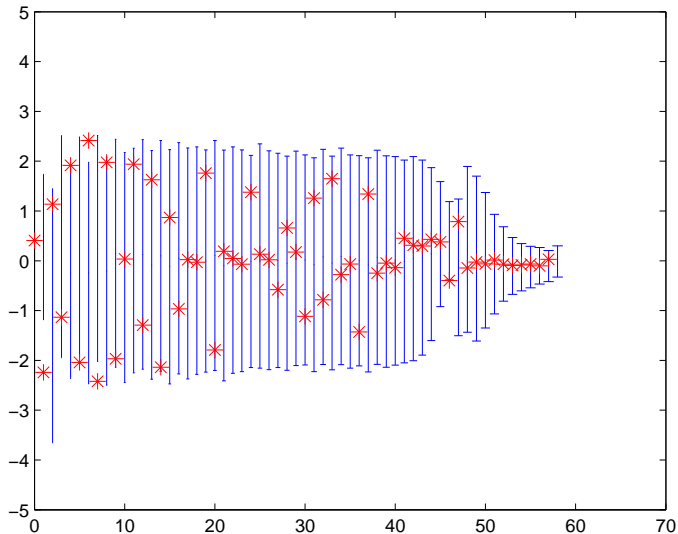
Motivating Example



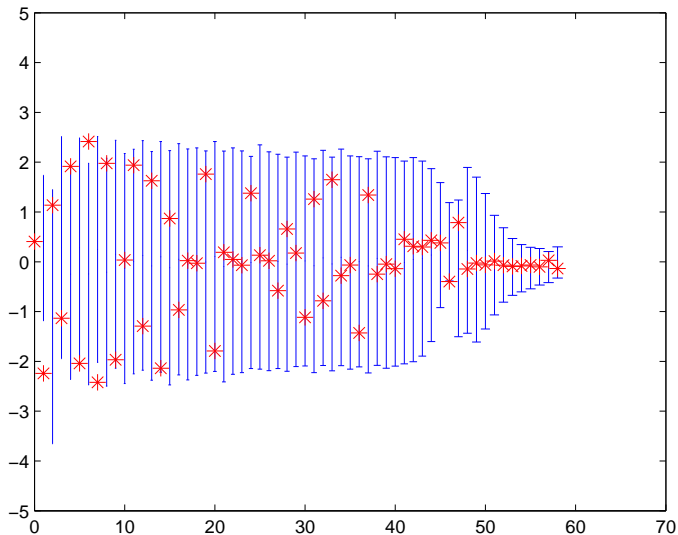
Motivating Example



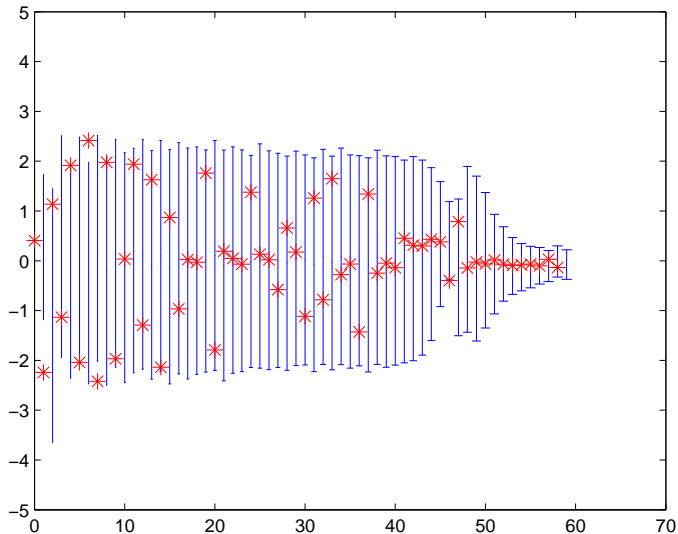
Motivating Example



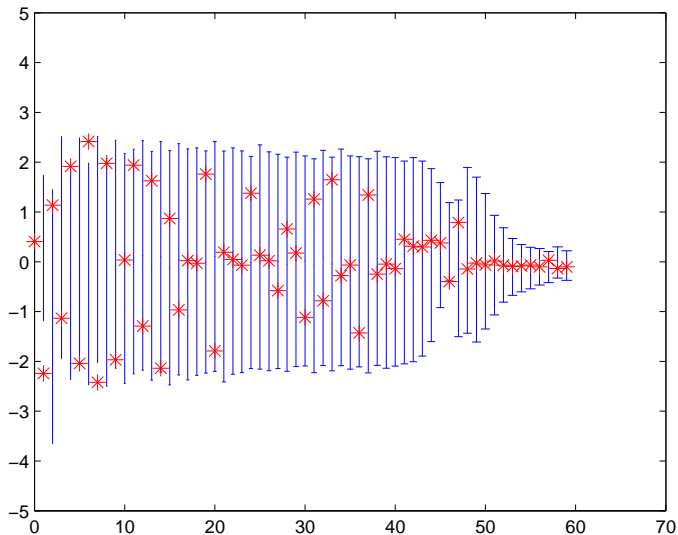
Motivating Example



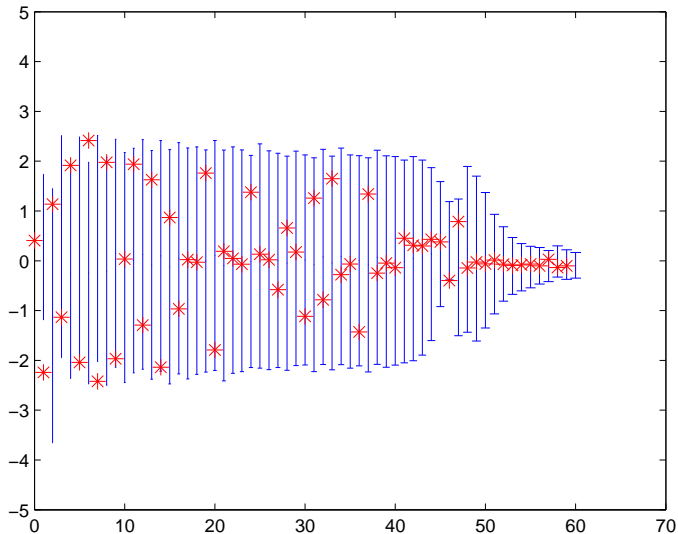
Motivating Example



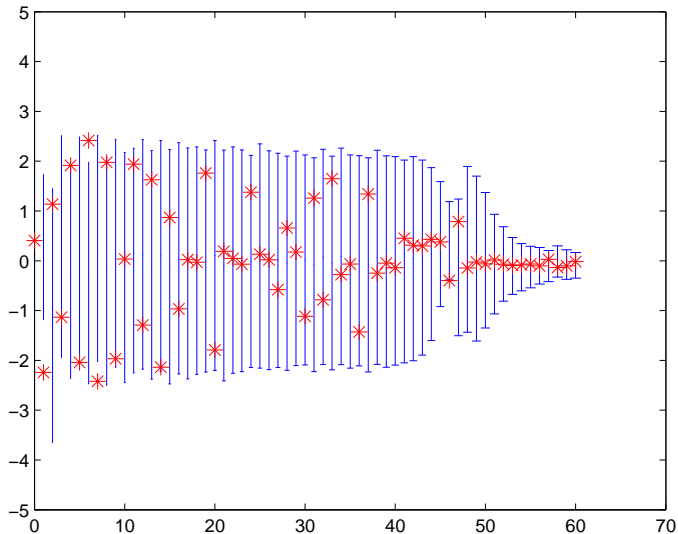
Motivating Example



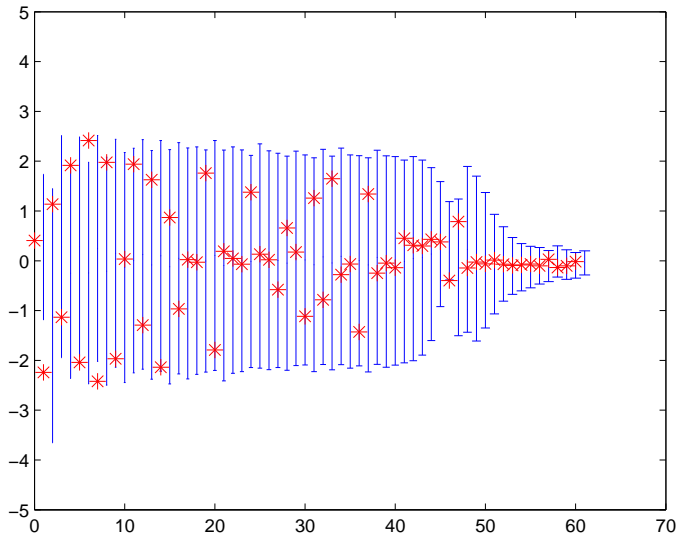
Motivating Example



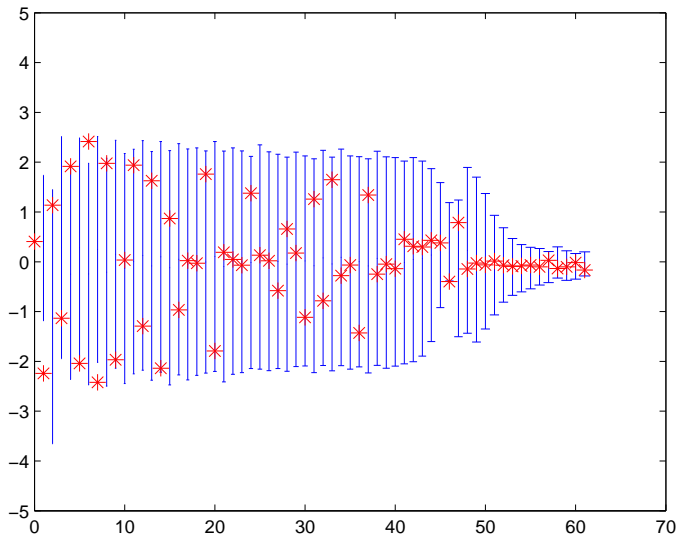
Motivating Example



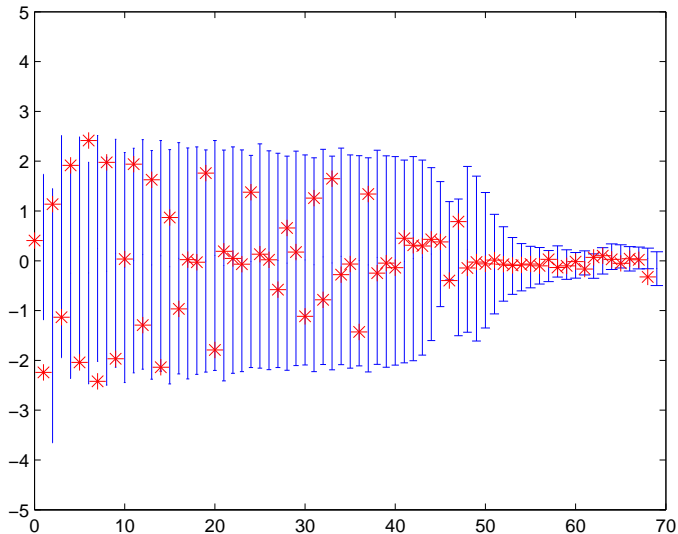
Motivating Example



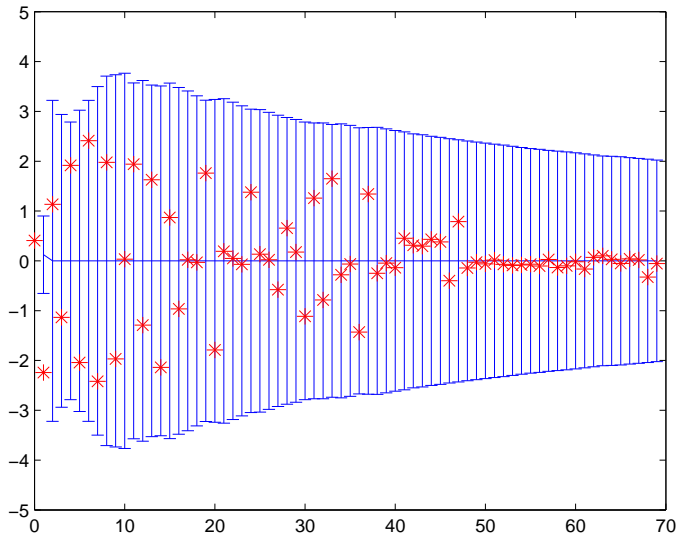
Motivating Example



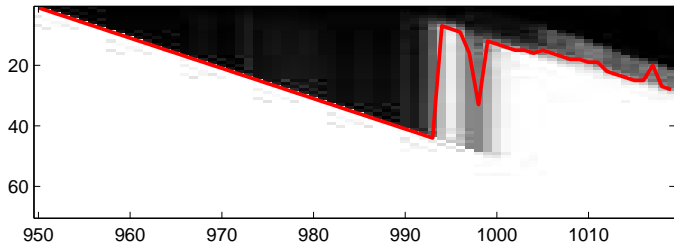
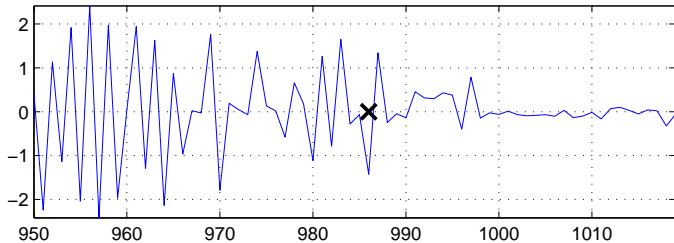
Motivating Example



Without Modelling Nonstationarity



Under the hood



BOCPD Algorithm

Predictions are made robust to change points as follows:

$$p(x_{t+1}|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_{1:t}, r_t)p(r_t|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_t^{(r)})p(r_t|x_{1:t})$$

BOCPD Algorithm

Predictions are made robust to change points as follows:

$$p(x_{t+1}|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_{1:t}, r_t)p(r_t|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_t^{(r)})p(r_t|x_{1:t})$$

$$\gamma_t := p(r_t, x_{1:t}) = \sum_{r_{t-1}} p(r_t, r_{t-1}, x_{1:t})$$

BOCPD Algorithm

Predictions are made robust to change points as follows:

$$p(x_{t+1}|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_{1:t}, r_t)p(r_t|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_t^{(r)})p(r_t|x_{1:t})$$

$$\begin{aligned}\gamma_t := p(r_t, x_{1:t}) &= \sum_{r_{t-1}} p(r_t, r_{t-1}, x_{1:t}) \\ &= \sum_{r_{t-1}} p(r_t, x_t | r_{t-1}, x_{1:t-1}) p(r_{t-1}, x_{1:t-1}) \\ &= \sum_{r_{t-1}} \underbrace{p(r_t | r_{t-1}, \theta_h)}_{\text{hazard}} \underbrace{p(x_t | r_{t-1}, x_t^{(r)}, \theta_m)}_{\text{likelihood (UPM)}} \underbrace{p(r_{t-1}, x_{1:t-1})}_{\gamma_{t-1}}\end{aligned}$$

BOCPD Algorithm

Predictions are made robust to change points as follows:

$$p(x_{t+1}|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_{1:t}, r_t)p(r_t|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_t^{(r)})p(r_t|x_{1:t})$$

$$\begin{aligned}\gamma_t := p(r_t, x_{1:t}) &= \sum_{r_{t-1}} p(r_t, r_{t-1}, x_{1:t}) \\ &= \sum_{r_{t-1}} p(r_t, x_t | r_{t-1}, x_{1:t-1}) p(r_{t-1}, x_{1:t-1}) \\ &= \sum_{r_{t-1}} \underbrace{p(r_t | r_{t-1}, \theta_h)}_{\text{hazard}} \underbrace{p(x_t | r_{t-1}, x_t^{(r)}, \theta_m)}_{\text{likelihood (UPM)}} \underbrace{p(r_{t-1}, x_{1:t-1})}_{\gamma_{t-1}}\end{aligned}$$

Defines forward **message passing** scheme.

θ_h and θ_m are the hazard function and UPM parameters.

Constitute hyperparameters of the overall model.

- Constant hazard function $p(r_t|r_{t-1}, \theta_h) := H(r|\theta_h) = \theta_h$

Simple Example

- Constant hazard function $p(r_t|r_{t-1}, \theta_h) := H(r|\theta_h) = \theta_h$
- Gives rise, *a priori*, to geometric inter-arrival times for change points.

Simple Example

- Constant hazard function $p(r_t|r_{t-1}, \theta_h) := H(r|\theta_h) = \theta_h$
- Gives rise, **a priori**, to geometric inter-arrival times for change points.
- Assume data in each segment is generated IID Normal, with Normal-Inverse-Gamma (NIG) prior.

Simple Example

- Constant hazard function $p(r_t|r_{t-1}, \theta_h) := H(r|\theta_h) = \theta_h$
- Gives rise, **a priori**, to geometric inter-arrival times for change points.
- Assume data in each segment is generated IID Normal, with Normal-Inverse-Gamma (NIG) prior.
- Thus the predictive distribution is a Student-t, and can be computed from the posterior NIG parameters.

Simple Example

- Constant hazard function $p(r_t|r_{t-1}, \theta_h) := H(r|\theta_h) = \theta_h$
- Gives rise, **a priori**, to geometric inter-arrival times for change points.
- Assume data in each segment is generated IID Normal, with Normal-Inverse-Gamma (NIG) prior.
- Thus the predictive distribution is a Student-t, and can be computed from the posterior NIG parameters.
- $p(x_{t+1}|x_{1:t}, r_t, \theta_m) = \text{Student-}t(\mu_r, \kappa_r, \alpha_r, \beta_r)$
e.g., $\alpha_r = \alpha_0 + r_t/2$
In this example $\theta_m := \{\mu_0, \kappa_0, \alpha_0, \beta_0\}$.

- Use a GP to define a distribution over **temporal** functions (GPTS).

- Use a GP to define a distribution over **temporal** functions (GPTS).
- A GP is characterized by a mean function and a covariance function $k(x_i, x_j)$.
- GP parameters θ_m control the mean and covariance functions.
- GPTS has many of the classical time series models as special cases (e.g. ARMA(p, q)).

- Use a GP to define a distribution over **temporal** functions (GPTS).
- A GP is characterized by a mean function and a covariance function $k(x_i, x_j)$.
- GP parameters θ_m control the mean and covariance functions.
- GPTS has many of the classical time series models as special cases (e.g. ARMA(p, q)).
- Next-step predictions are **Gaussian**: $p(x_t | x_{(t-r_t):(t-1)}, \theta_m) = \mathcal{N}(m_t, v_t)$.

- Use a GP to define a distribution over **temporal** functions (GPTS).
- A GP is characterized by a mean function and a covariance function $k(x_i, x_j)$.
- GP parameters θ_m control the mean and covariance functions.
- GPTS has many of the classical time series models as special cases (e.g. ARMA(p, q)).
- Next-step predictions are **Gaussian**: $p(x_t | x_{(t-r_t):(t-1)}, \theta_m) = \mathcal{N}(m_t, v_t)$.

$$m_t = \mathbf{k}_\star^\top (K + \sigma_n^2 I)^{-1} \mathbf{x},$$
$$v_t = k(x_t, x_t) - \mathbf{k}_\star^\top (K + \sigma_n^2 I)^{-1} \mathbf{k}_\star.$$

- Use a GP to define a distribution over **temporal** functions (GPTS).
- A GP is characterized by a mean function and a covariance function $k(x_i, x_j)$.
- GP parameters θ_m control the mean and covariance functions.
- GPTS has many of the classical time series models as special cases (e.g. ARMA(p, q)).
- Next-step predictions are **Gaussian**: $p(x_t | x_{(t-r_t):(t-1)}, \theta_m) = \mathcal{N}(m_t, v_t)$.

$$m_t = \mathbf{k}_\star^\top (K + \sigma_n^2 I)^{-1} \mathbf{x},$$
$$v_t = k(x_t, x_t) - \mathbf{k}_\star^\top (K + \sigma_n^2 I)^{-1} \mathbf{k}_\star.$$

- Example covariance function $k(x_i, x_j) = \sigma_f^2 \exp(-\frac{(x_i - x_j)^2}{2\ell^2})$.
- In this case $\theta_m = \{\ell, \sigma_f, \sigma_n\}$.

- Problem with GPTS: GP hyperparameters are the UPM parameters. We cannot model changes in these.

- Problem with GPTS: GP hyperparameters are the UPM parameters. We cannot model changes in these.
- Solution: UPM needs to [integrate out](#) the GP hyperparameters!

Gaussian Process UPMs II — NSGP

- Problem with GPTS: GP hyperparameters are the UPM parameters. We cannot model changes in these.
- Solution: UPM needs to **integrate out** the GP hyperparameters!
- θ_m now parameterizes the prior over GP hyperparameters:

$$p(x_t|\mathbf{x}, \theta_m) = \int p(x_t|\mathbf{x}, \lambda)p(\lambda|\mathbf{x}) d\lambda$$

- Problem with GPTS: GP hyperparameters are the UPM parameters. We cannot model changes in these.
- Solution: UPM needs to **integrate out** the GP hyperparameters!
- θ_m now parameterizes the prior over GP hyperparameters:

$$\begin{aligned} p(x_t|\mathbf{x}, \theta_m) &= \int p(x_t|\mathbf{x}, \lambda)p(\lambda|\mathbf{x}) d\lambda \\ &= \frac{1}{Z} \int p(x_t|\mathbf{x}, \lambda)p(\mathbf{x}|\lambda)p(\lambda|\theta_m) d\lambda \end{aligned}$$

- Problem with GPTS: GP hyperparameters are the UPM parameters. We cannot model changes in these.
- Solution: UPM needs to **integrate out** the GP hyperparameters!
- θ_m now parameterizes the prior over GP hyperparameters:

$$\begin{aligned} p(x_t|\mathbf{x}, \theta_m) &= \int p(x_t|\mathbf{x}, \lambda)p(\lambda|\mathbf{x}) d\lambda \\ &= \frac{1}{Z} \int p(x_t|\mathbf{x}, \lambda)p(\mathbf{x}|\lambda)p(\lambda|\theta_m) d\lambda \end{aligned}$$

λ are the usual GP hypers, e.g., $\lambda := \{\ell, \sigma_f, \sigma_n\}$.

- Problem with GPTS: GP hyperparameters are the UPM parameters. We cannot model changes in these.
- Solution: UPM needs to **integrate out** the GP hyperparameters!
- θ_m now parameterizes the prior over GP hyperparameters:

$$\begin{aligned} p(x_t|\mathbf{x}, \theta_m) &= \int p(x_t|\mathbf{x}, \lambda)p(\lambda|\mathbf{x}) d\lambda \\ &= \frac{1}{Z} \int p(x_t|\mathbf{x}, \lambda)p(\mathbf{x}|\lambda)p(\lambda|\theta_m) d\lambda \end{aligned}$$

λ are the usual GP hypers, e.g., $\lambda := \{\ell, \sigma_f, \sigma_n\}$.

$p(\mathbf{x}|\lambda)$ is known as the marginal likelihood of the GP:

$$2\pi^{-r_t/2} |K + \sigma_n^2 I|^{-1/2} \exp\left(-\frac{1}{2} \mathbf{x}^\top (K + \sigma_n^2 I)^{-1} \mathbf{x}\right)$$

Gaussian Process UPMs II — NSGP (Approx. Inference)

- Marginal likelihood above is a nonlinear function of λ .
- Required integrals are therefore always intractable.

Gaussian Process UPMs II — NSGP (Approx. Inference)

- Marginal likelihood above is a nonlinear function of λ .
- Required integrals are therefore always intractable.
- If number of GP hyperparameters is low, can use grid method:

Gaussian Process UPMs II — NSGP (Approx. Inference)

- Marginal likelihood above is a nonlinear function of λ .
- Required integrals are therefore always intractable.
- If number of GP hyperparameters is low, can use grid method:

$$p(x_t | \mathbf{x}, \theta_m) \approx \sum_{\lambda_g} p(x_t | \mathbf{x}, \lambda_g) \left(\frac{p(\mathbf{x} | \lambda_g)}{\sum_{\lambda_g} p(\mathbf{x} | \lambda_g)} \right).$$

- More sophisticated quadrature is tricky: integrand is always positive

Gaussian Process UPMs II — NSGP (Approx. Inference)

- Marginal likelihood above is a nonlinear function of λ .
- Required integrals are therefore always intractable.
- If number of GP hyperparameters is low, can use grid method:

$$p(x_t | \mathbf{x}, \theta_m) \approx \sum_{\lambda_g} p(x_t | \mathbf{x}, \lambda_g) \left(\frac{p(\mathbf{x} | \lambda_g)}{\sum_{\lambda_g} p(\mathbf{x} | \lambda_g)} \right).$$

- More sophisticated quadrature is tricky: integrand is always positive
- For higher dimensional problem use Hamiltonian Monte Carlo (HMC)

$$p(x_t | \mathbf{x}, \theta_m) \approx \sum_{\lambda_s} p(x_t | \mathbf{x}, \lambda_s).$$

- The posterior $p(\lambda | x_{(t-r_t):(t-1)})$, represented by samples $\{\lambda_s^{(t-1)}\}$, will look similar to $p(\lambda | x_{(t-\tau):t})$.

Gaussian Process UPMs II — NSGP (Approx. Inference)

- Marginal likelihood above is a nonlinear function of λ .
- Required integrals are therefore always intractable.
- If number of GP hyperparameters is low, can use grid method:

$$p(x_t | \mathbf{x}, \theta_m) \approx \sum_{\lambda_g} p(x_t | \mathbf{x}, \lambda_g) \left(\frac{p(\mathbf{x} | \lambda_g)}{\sum_{\lambda_g} p(\mathbf{x} | \lambda_g)} \right).$$

- More sophisticated quadrature is tricky: integrand is always positive
- For higher dimensional problem use Hamiltonian Monte Carlo (HMC)

$$p(x_t | \mathbf{x}, \theta_m) \approx \sum_{\lambda_s} p(x_t | \mathbf{x}, \lambda_s).$$

- The posterior $p(\lambda | x_{(t-r_t):(t-1)})$, represented by samples $\{\lambda_s^{(t-1)}\}$, will look similar to $p(\lambda | x_{(t-\tau):t})$.
- Initialize the HMC sampler at $\{\lambda_s^{(t-1)}\}$ and run for a short number of iterations for each sample.

- Implement $p(x_t|x_{t-r_t}, \dots, x_{t-1}, \theta_m)$ **directly** using a GP.
- θ_m are the standard GP hyperparameters in this case.

- Implement $p(x_t|x_{t-r_t}, \dots, x_{t-1}, \theta_m)$ **directly** using a GP.
- θ_m are the standard GP hyperparameters in this case.

$$x_t = f(x_{t-r_t:t-1}) + \epsilon_t,$$
$$f \sim \mathcal{GP}(0, K), \epsilon_t \sim \mathcal{N}(0, \sigma^2).$$

- Implement $p(x_t|x_{t-r_t}, \dots, x_{t-1}, \theta_m)$ **directly** using a GP.
- θ_m are the standard GP hyperparameters in this case.

$$x_t = f(x_{t-r_t:t-1}) + \epsilon_t,$$
$$f \sim \mathcal{GP}(0, K), \epsilon_t \sim \mathcal{N}(0, \sigma^2).$$

- The ARGP can have more complex dynamics than the GPTS, but is restricted to the discrete time domain.

- If ideas in this talk are implemented in the most naive way complexity can be as high as $\mathcal{O}(T^5)$!

Improving Runtime

- If ideas in this talk are implemented in the most naive way complexity can be as high as $\mathcal{O}(T^5)$!
- **Pruning**: Remove all run-length hypotheses with posterior mass below a certain threshold.

- If ideas in this talk are implemented in the most naive way complexity can be as high as $\mathcal{O}(T^5)$!
- **Pruning**: Remove all run-length hypotheses with posterior mass below a certain threshold.
- ... or keep track of the K -best.

- If ideas in this talk are implemented in the most naive way complexity can be as high as $\mathcal{O}(T^5)$!
- **Pruning**: Remove all run-length hypotheses with posterior mass below a certain threshold.
- ... or keep track of the K -best.
- Use **rank-1** updates when augmenting the GP model with a fresh observation.

- If ideas in this talk are implemented in the most naive way complexity can be as high as $\mathcal{O}(T^5)$!
- **Pruning**: Remove all run-length hypotheses with posterior mass below a certain threshold.
- ... or keep track of the K -best.
- Use **rank-1** updates when augmenting the GP model with a fresh observation.
- If operating in discrete-time, use **Toeplitz matrix** methods (Yule-Walker recursion is a sequential algorithm).

- If ideas in this talk are implemented in the most naive way complexity can be as high as $\mathcal{O}(T^5)$!
- **Pruning**: Remove all run-length hypotheses with posterior mass below a certain threshold.
- ... or keep track of the K -best.
- Use **rank-1** updates when augmenting the GP model with a fresh observation.
- If operating in discrete-time, use **Toeplitz matrix** methods (Yule-Walker recursion is a sequential algorithm).
- Overall, can bring down complexity to $\mathcal{O}(T\tilde{R}^2/\epsilon)$ or $\mathcal{O}(TK\tilde{R}^2)$.

- If ideas in this talk are implemented in the most naive way complexity can be as high as $\mathcal{O}(T^5)$!
- **Pruning**: Remove all run-length hypotheses with posterior mass below a certain threshold.
- ... or keep track of the K -best.
- Use **rank-1** updates when augmenting the GP model with a fresh observation.
- If operating in discrete-time, use **Toeplitz matrix** methods (Yule-Walker recursion is a sequential algorithm).
- Overall, can bring down complexity to $\mathcal{O}(T\tilde{R}^2/\epsilon)$ or $\mathcal{O}(TK\tilde{R}^2)$.
- $\mathcal{O}(T\tilde{R}/\epsilon)$ or $\mathcal{O}(TK\tilde{R})$ if using Toeplitz methods.

Learning BOCPD Hyperparameters

Recall:

$$p(x_{t+1}|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_{1:t}, r_t)p(r_t|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_t^{(r)})p(r_t|x_{1:t})$$

Learning BOCPD Hyperparameters

Recall:

$$p(x_{t+1}|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_{1:t}, r_t)p(r_t|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_t^{(r)})p(r_t|x_{1:t})$$

$$\gamma_t := p(r_t, x_{1:t}) = \sum_{r_{t-1}} p(r_t, r_{t-1}, x_{1:t})$$

Learning BOCPD Hyperparameters

Recall:

$$p(x_{t+1}|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_{1:t}, r_t)p(r_t|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_t^{(r)})p(r_t|x_{1:t})$$

$$\begin{aligned}\gamma_t := p(r_t, x_{1:t}) &= \sum_{r_{t-1}} p(r_t, r_{t-1}, x_{1:t}) \\ &= \sum_{r_{t-1}} p(r_t, x_t | r_{t-1}, x_{1:t-1}) p(r_{t-1}, x_{1:t-1}) \\ &= \sum_{r_{t-1}} \underbrace{p(r_t | r_{t-1}, \theta_h)}_{\text{hazard}} \underbrace{p(x_t | r_{t-1}, x_t^{(r)}, \theta_m)}_{\text{likelihood (UPM)}} \underbrace{p(r_{t-1}, x_{1:t-1})}_{\gamma_{t-1}}\end{aligned}$$

Learning BOCPD Hyperparameters

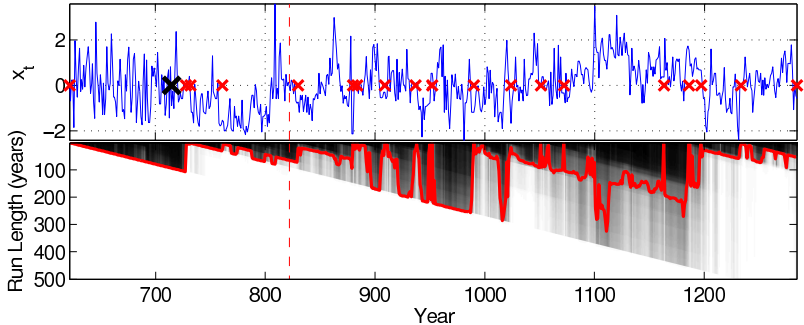
Recall:

$$p(x_{t+1}|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_{1:t}, r_t)p(r_t|x_{1:t}) = \sum_{r_t} p(x_{t+1}|x_t^{(r)})p(r_t|x_{1:t})$$

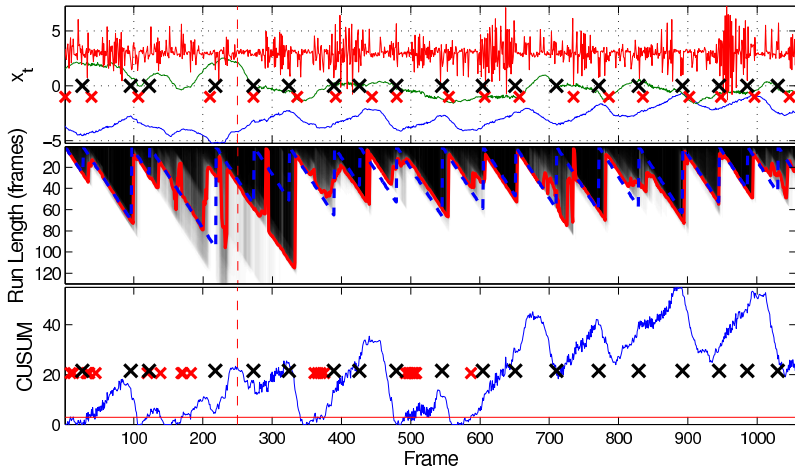
$$\begin{aligned}\gamma_t := p(r_t, x_{1:t}) &= \sum_{r_{t-1}} p(r_t, r_{t-1}, x_{1:t}) \\ &= \sum_{r_{t-1}} p(r_t, x_t | r_{t-1}, x_{1:t-1}) p(r_{t-1}, x_{1:t-1}) \\ &= \sum_{r_{t-1}} \underbrace{p(r_t | r_{t-1}, \theta_h)}_{\text{hazard}} \underbrace{p(x_t | r_{t-1}, x_t^{(r)}, \theta_m)}_{\text{likelihood (UPM)}} \underbrace{p(r_{t-1}, x_{1:t-1})}_{\gamma_{t-1}}\end{aligned}$$

- Can differentiate the above expression w.r.t. θ_m and θ_h .
- Boils down to computing $\frac{\partial}{\partial \theta_m} p(x_t | r_{t-1}, x_t^{(r)}, \theta_m)$, and, $\frac{\partial}{\partial \theta_h} p(r_t | r_{t-1}, \theta_h)$.
- Allows for online updating of BOCPD hyperparameters.

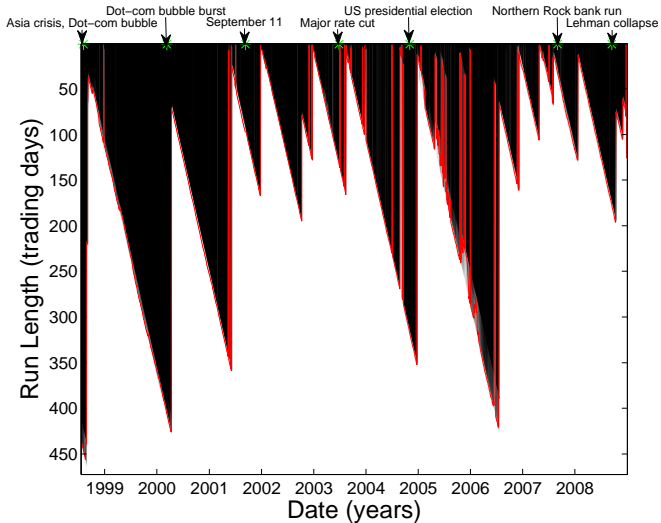
Results: Nile Data



Results: Bee Waggle-Dance Data



Results: Industry Portfolio Data



Results: Summary

Method	Negative Log Likelihood	p-value	MSE	p-value
Nile Data (200 Training Points, 462 Test Points)				
GPTS	1.19±0.0548	0.196	0.579±0.0976	0.356
♥ GPTS-CP	1.19±0.0548	0.167	0.583±0.0989	0.335
ARGP	1.18±0.0510	0.202	0.568±0.0940	0.410
♥ ARGP-CP	1.15 ± 0.0555	N/A	0.553 ± 0.0962	N/A
Kalman	1.17±0.0508	0.361	0.562±0.121	0.453
TIM	1.49±0.0714	<0.001	1.16±0.161	<0.001
♥ NSGP (grid)	1.15±0.0655	0.490	0.585±0.0988	0.321
Bee Waggle Dance Data (250 Training Points, 807 Test Points)				
GPTS	8.02±0.504	<0.001	8.44±0.745	<0.001
♥ GPTS-CP	4.54±0.188	<0.001	3.13±0.241	<0.001
ARGP	4.35±0.167	0.007	2.98±0.224	0.008
♥ ARGP-CP	4.07 ± 0.150	N/A	2.62 ± 0.195	N/A
Kalman	4.39±0.176	0.002	2.93±0.215	0.016
TIM	4.54±0.177	<0.001	3.25±0.237	<0.001
♥ NSGP (HMC)	4.19±0.212	<0.001	3.17±0.230	<0.001