Probabilistic Fitting

Probabilistic Morphable Models

Summer School, June 2017

Sandro Schönborn

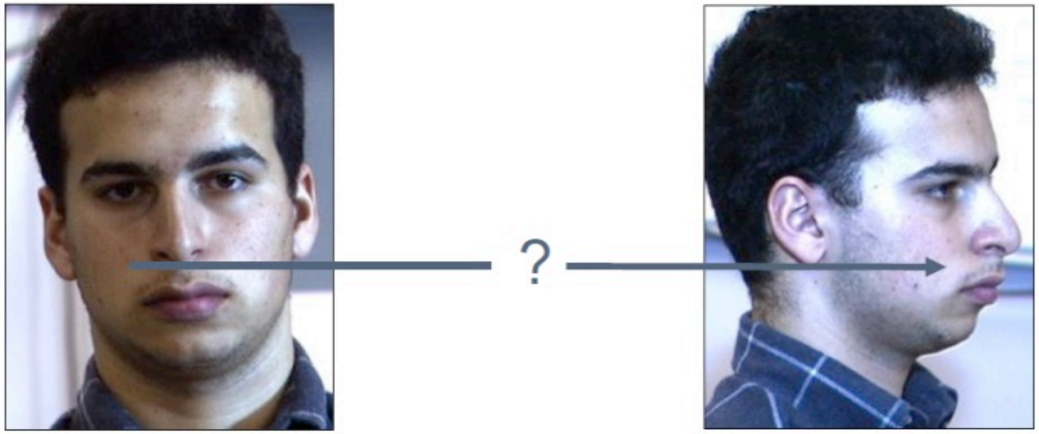
University of Basel

Probabilistic Inference for Face Model Fitting

Approximate Inference with Markov Chain Monte Carlo

Probabilistic Registration

- Model-based face image registration



- Probabilistic Gaussian Process framework
- *Bayesian Fitting framework*

Images: CMU PIE Database
Sim, Baker, Bsat, 2002

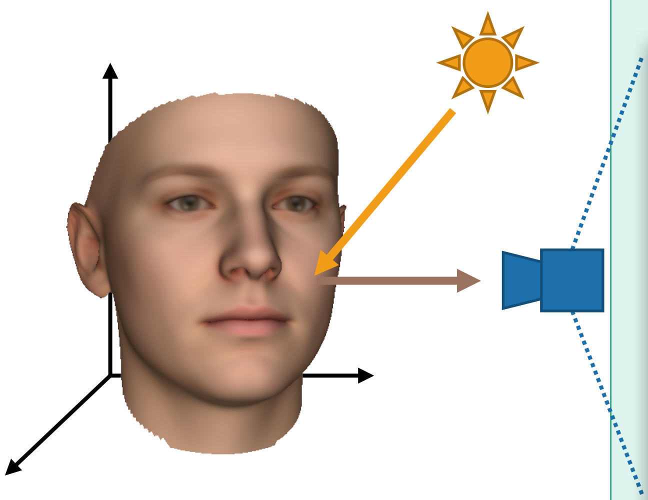
Face Image Manipulation

perceived as more *trustworthy*

3D Face Reconstruction

Concept: 3D Face Model Fitting

Face Model

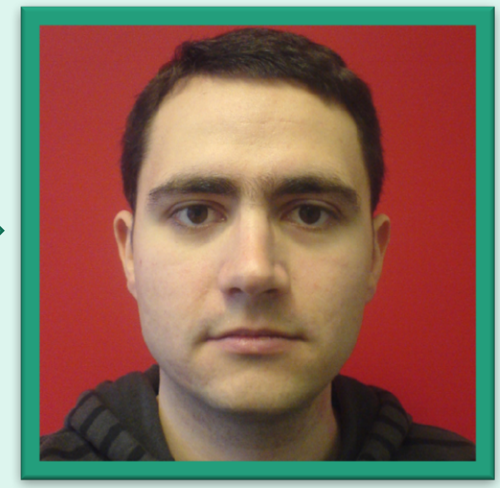


Parametric face model

Reconstruction: Analysis-by-Synthesis

Rendered Image $I(\theta)$

Target Image I



Likelihood $\ell(\theta; I) \propto P(I \mid I(\theta))$

$\theta = (\vartheta, \alpha, \beta, l)$: ϑ Scene Parameters, α Face shape, β Face color, l Illumination

Formal: 3D Face Model Fitting

- 3D face model: $(I_R + h_C) \circ h_S$
  - Color model: $I_R + h_C$
  - Shape model: $I_R \circ h_S$
- 3D-2D computer graphics:
  - $\boldsymbol{x}^{2D} = \mathrm{T_{IMG}}\left(\mathrm{Pr}\left(\mathrm{T_{3D}}\left(\boldsymbol{S}(\boldsymbol{x}^{3D})\right)\right)\right)$
    - Rigid 3D $\mathrm{T_{3D}}$, transform in image $\mathrm{T_{IMG}}$
    - Projection $\mathrm{Pr}(\boldsymbol{x}) = \begin{bmatrix} x/z \\ y/z \end{bmatrix}$
  - $\boldsymbol{I}(\boldsymbol{x}^{2D}) = \mathrm{C_T}\left(\mathrm{L}\left(\boldsymbol{n}(\boldsymbol{x}^{3D}), \boldsymbol{C}(\boldsymbol{x}^{3D}), \boldsymbol{x}^{3D}\right)\right)$
    - Normal $\boldsymbol{n}$, Color transform $\mathrm{C_T}(\boldsymbol{c})$, illumination $\mathrm{L}(\boldsymbol{n}, \boldsymbol{c}, \boldsymbol{x})$

Corresponding $\boldsymbol{x}^{2D}$ and $\boldsymbol{x}^{3D}$

Overview

- Computer Graphics Overview
- Probabilistic Setup
- **Markov Chain Monte Carlo**
  - **Markov Chains**
- **3D Fitting Problem**
  - **Landmarks**
- 2D Face Image Analysis
  - Image fitting
  - Filtering with unreliable information

Approximate Bayesian Inference with Samples

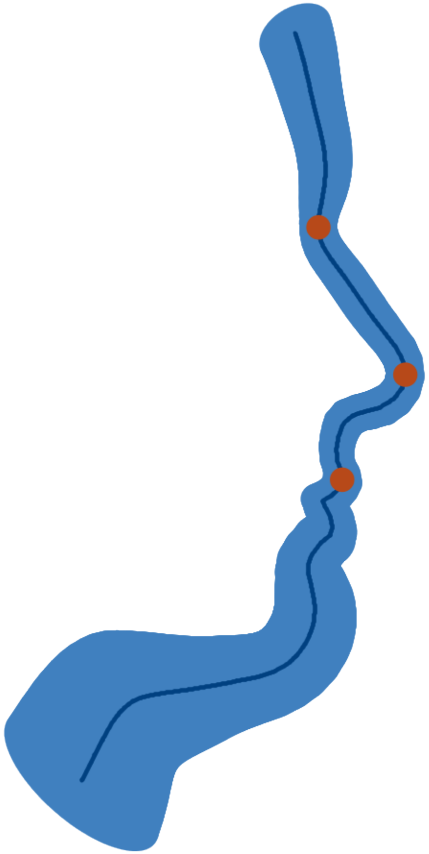
Simulating the Posterior Distribution

Reminder: General Bayesian Inference

- Observation of *additional* variables
  - Common case, e.g. face rendering, landmark locations
  - Coupled to core model via likelihood factorization
- General Bayesian inference case:
  - Distribution of data D (formerly Evidence)
  - Parameters θ (formerly Query)

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

$$P(\theta|D) \propto P(D|\theta)P(\theta)$$



Data: our *image* or *landmarks*, etc.
Model: shape and color model of faces, 3d graphics scene

Bayesian Inference and Estimation

- Bayes
  - Whole posterior *distribution*
  - Belief update (Bayes rule, Bayesian inference)
  - Captures uncertainty

$$p(\theta|D) = \frac{\ell(\theta;D)p(\theta)}{\int \ell(\theta;D)p(\theta)\mathrm{d}\theta}$$

Approximation

- Maximum-A-Posteriori (MAP):
  - Single value
  - Maximum of *posterior* distribution – "regularized"

$$\hat{\theta} = \arg\max_{\theta} \ell(\theta;D)p(\theta)$$

- Maximum Likelihood (ML):
  - Single value
  - Maximum of *likelihood* only

$$\hat{\theta} = \arg\max_{\theta} \ell(\theta;D)$$

$$\ell(\theta;D) = P(D|\theta)$$

Bayesian Fitting

- Posterior distribution

$$p(\alpha|I_T, M) = \frac{p(\alpha)p(I_T|\alpha, M)}{N(I_T; M)}$$

- Prior deformations of the mean face: $p(\varphi)$

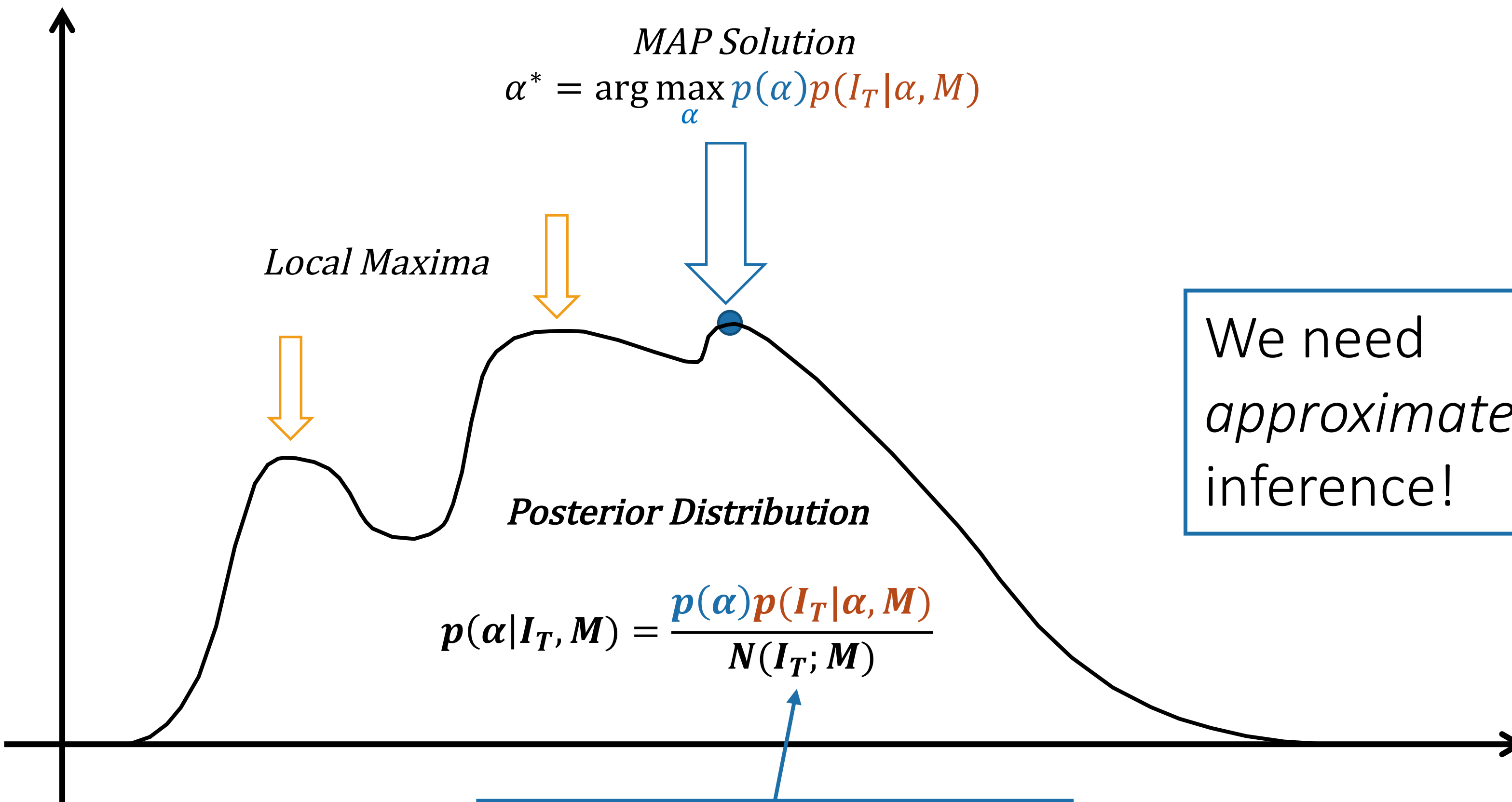
$$\varphi \sim GP(\mu, k){:}\ \ \varphi \approx M[\alpha] = \mu + \sum_{i}^{d} \alpha_i \sqrt{\lambda_i} \Phi_i$$

parameterization: low-rank models

$$\alpha \sim N(0, E_d)$$

- Likelihood, e.g. $p(I_T|\alpha, I_R) \propto \exp \frac{-D[I_T, I_R \circ M[\alpha]]}{\sigma^2}$

Posterior distribution



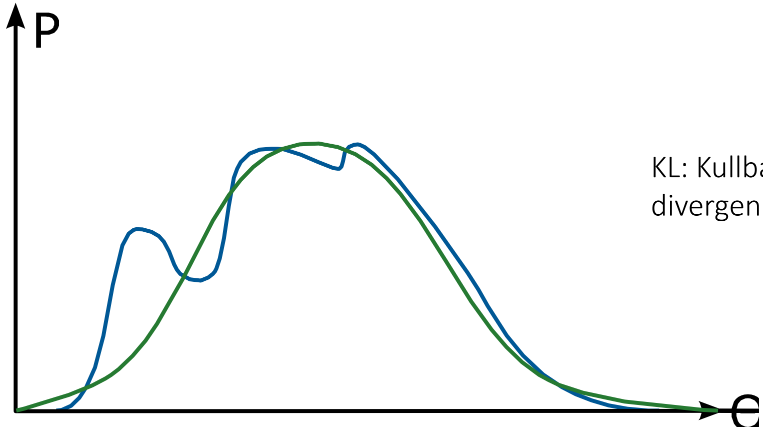
We need *approximate* inference!

Infeasible to compute:
$N(I_T) = \int p(\alpha)p(I_T|\alpha, M)d\alpha$

Approximate Bayesian Inference

Variational methods

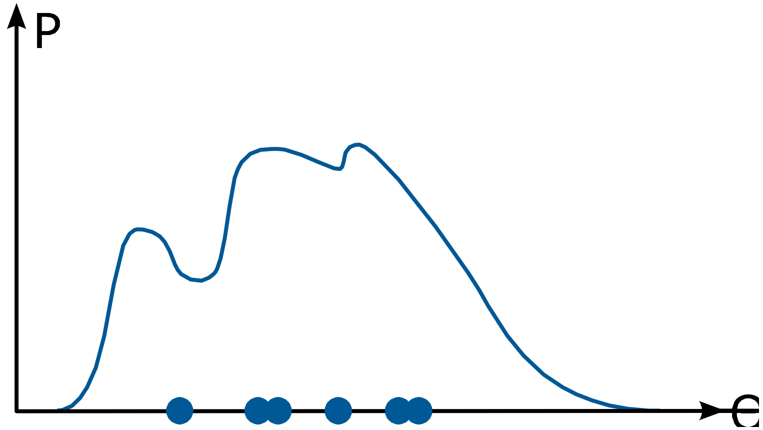
- Function approximation $q(\theta)$
  $$\arg\max_q \mathrm{KL}(q(\theta)|p(\theta|D))$$
- Variational Message Passing, Mean-Field Theory, Moment matching, …



KL: Kullback-Leibler divergence

Sampling methods

- Numeric approximations through simulation
- Monte Carlo, Importance sampling, Particle Filters, MCMC, …



Sampling Methods

- Simulate a distribution p through random samples x_i
- Evaluate expectations

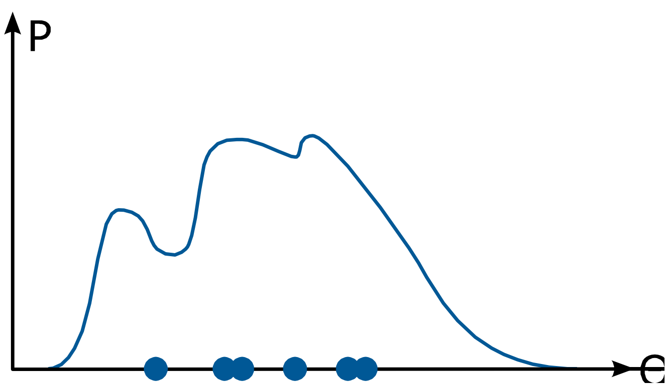
$$E[f(x)] = \int f(x)p(x)dx$$

$$E[f(x)] \approx \hat{f} = \frac{1}{N}\sum_{i}^{N} f(x_i), \qquad x_i \sim p(x)$$

This is difficult!

$$V[\hat{f}] \sim O\left(\frac{1}{N}\right)$$

- “Independent” of dimensionality
- More samples increase accuracy



Sampling from A Distribution

- Easy for standard distributions … is it?
  - Uniform
  - Gaussian

```
Random.nextDouble()
Random.nextGaussian()
```

- How to sample from more complex distributions?
  - Beta, Exponential, Chi square, Gamma, …
  - Posteriors are very often not in a “nice” standard text book form
- Sadly, only very few distributions are easy to sample from
  - We need to sample from an unknown posterior with only *unnormalized, expensive point-wise evaluation* ☹
- General Samplers?
  - Yes! – Rejection, Importance, *MCMC*

Markov Chain Monte Carlo

- Markov Chain Monte Carlo Methods (MCMC)

  Design a *Markov Chain* such that samples x obey the target distribution p

  Concept: *"Use an already existing sample to produce the next one"*
- Very powerful general sampling methods
  - Many successful practical applications
  - Proven: developed in the 1950/1970ies (Metropolis/Hastings)
  - Direct mapping of computing power to approximation accuracy
- Algorithms (buzz words):
  - Metropolis/-Hastings, Gibbs, Slice Sampling

Markov Chains

Understanding Markov Chain Monte Carlo Methods

Markov Chain

- Sequence of random variables $\{X_i\}_{i=1}^{N},\ X_i \in S$ with joint distribution

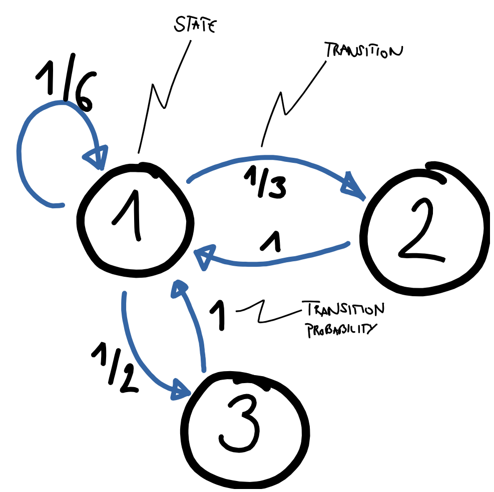
State space

$$P(X_1, X_2, \dots, X_N) = P(X_1) \prod_{i=2}^{N} P(X_i | X_{i-1})$$

Initial distribution

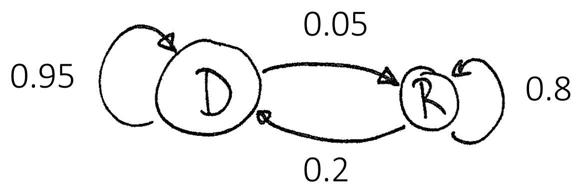
Transition probability

- Simplifications: (for our analysis)
  - Discrete state space: $S = \{1, 2, \dots, K\}$
  - Homogeneous Chain: $P(X_i = l | X_{i-1} = m) = T_{lm}$



Example: Markov Chain

- Simple weather model: *dry* (D) or *rainy* (R) hour
  - Condition in next hour? X_{t+1}
  - State space $S = \{D, R\}$
  - Stochastic: $P(X_{t+1}|X_t)$
  - Depends only on *current* condition X_t



- Draw Samples from chain:
  - Initial: $X_0 = D$
  - Evolution: $P(X_{t+1}|X_t)$

$DDDDDDDDRRRRRRRRRRRDDDDDDDDDDD$
$DDDDDDDDDDDDDDDDDDDDDDDDDDDDDD$
$DDDDDDDDDRDD...$

- Long-term Behavior
  - Does it converge? *Average* probability of rain?
  - Dynamics? How *quickly* will it converge?

Discrete Homogeneous Markov Chain

Formally linear algebra:

- Distribution (vector):

$$P(X_i):\ \boldsymbol{p_i} = \begin{bmatrix} P(X_i = 1) \\ \vdots \\ P(X_i = K) \end{bmatrix}$$

- Transition probability (transition matrix):

$$P(X_i|X_{i-1}):\ T = \begin{bmatrix} P(1 \leftarrow 1) & \cdots & P(1 \leftarrow K) \\ \vdots & \ddots & \vdots \\ P(K \leftarrow 1) & \cdots & P(K \leftarrow K) \end{bmatrix}$$

$$T_{lm} = P(l \leftarrow m) = P(X_i = l|X_{i-1} = m)$$

Evolution of the Initial Distribution

- Evolution of $P(X_1) \to P(X_2)$:

$$P(X_2 = l) = \sum_{m \in S} P(l \leftarrow m) P(X_1 = m)$$

$$\boldsymbol{p}_2 = T\boldsymbol{p}_1$$

- Evolution of n steps:

$$\boldsymbol{p}_{n+1} = T^n \boldsymbol{p}_1$$

- Is there a *stable* distribution $\boldsymbol{p}^*$? (steady-state)

$$\boldsymbol{p}^* = T\boldsymbol{p}^*$$

A stable distribution is an *eigenvector* of T with eigenvalue $\lambda = 1$

Steady-State Distribution: $\boldsymbol{p}^*$

- It exists:
  - T subject to normalization constraint: *left* eigenvector to eigenvalue 1
  $$\sum_l T_{lm} = 1 \quad \Leftrightarrow \quad [1 \quad \dots \quad 1]T = [1 \quad \dots \quad 1]$$
  - T has eigenvalue $\lambda = 1$ (left-/right eigenvalues are the same)
  - Steady-state distribution as corresponding right eigenvector
  $$T\boldsymbol{p}^* = \boldsymbol{p}^*$$
- Does *any* arbitrary initial distribution *evolve* to $\boldsymbol{p}^*$?
  - Convergence?
  - Uniqueness?

Equilibrium Distribution: $\boldsymbol{p}^*$

- Additional requirement for T: $(T_{lm})^n > 0$ for $n > N_0$

  The chain is called *irreducible* and *aperiodic* (implies *ergodic*)
  - All states are connected using at most N_0 steps
  - Return intervals to a certain state are irregular
- *Perron-Frobenius* theorem for positive matrices:
  - PF1: $\lambda_1 = 1$ is a simple eigenvalue with 1d eigenspace (*uniqueness*)
  - PF2: $\lambda_1 = 1$ is dominant, all $|\lambda_i| < 1, \quad i \neq 1$ (*convergence*)
- $\boldsymbol{p}^*$ is a stable attractor, called *equilibrium distribution*

$$T\boldsymbol{p}^* = \boldsymbol{p}^*$$

Convergence

- Time evolution of arbitrary distribution $\boldsymbol{p}_0$

$$\boldsymbol{p}_n = T^n \boldsymbol{p}_0$$

- Expand $\boldsymbol{p}_0$ in Eigen basis of T:

$$T\boldsymbol{e}_i = \lambda_i \boldsymbol{e}_i, \qquad |\lambda_i| < \lambda_1 = 1, \qquad |\lambda_k| \geq |\lambda_{k+1}|$$

$$\boldsymbol{p}_0 = \sum_i^K c_i \boldsymbol{e}_i$$

$$T\boldsymbol{p}_0 = \sum_i^K c_i \lambda_i \boldsymbol{e}_i$$

$$T^n \boldsymbol{p}_0 = \sum_i^K c_i \lambda_i^n \boldsymbol{e}_i = c_1 \boldsymbol{e}_1 + \lambda_2^n c_2 \boldsymbol{e}_2 + \lambda_3^n c_3 \boldsymbol{e}_3 + \cdots$$

Convergence (II)

$$T^n \boldsymbol{p}_0 = \sum_i^K c_i \lambda_i^n \boldsymbol{e}_i = c_1 \boldsymbol{e}_1 + \lambda_2^n c_2 \boldsymbol{e}_2 + \lambda_3^n c_3 \boldsymbol{e}_3 + \cdots$$

$$(n \gg 1) \qquad \approx \boldsymbol{p}^* + \lambda_2^n c_2 \boldsymbol{e}_2$$

$c_1 \boldsymbol{e}_1 = \boldsymbol{p}^*$

Normalizations:
$\|\boldsymbol{e}_1\| = 1$
$\sum_i p_i^* = 1$

- We have *convergence*:

$$T^n \boldsymbol{p}_0 \xrightarrow{n \to \infty} \boldsymbol{p}^*$$

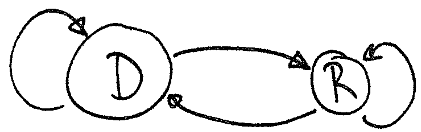
- *Rate* of convergence:

$$\|\boldsymbol{p}_n - \boldsymbol{p}^*\| \approx \|\lambda_2^n c_2 \boldsymbol{e}_2\| = |\lambda_2|^n |c_2|$$

Example: Weather Dynamics

Rain forecast for stable versus mixed weather:

stable $W_s = \begin{bmatrix} 0.95 & 0.2 \\ 0.05 & 0.8 \end{bmatrix}$



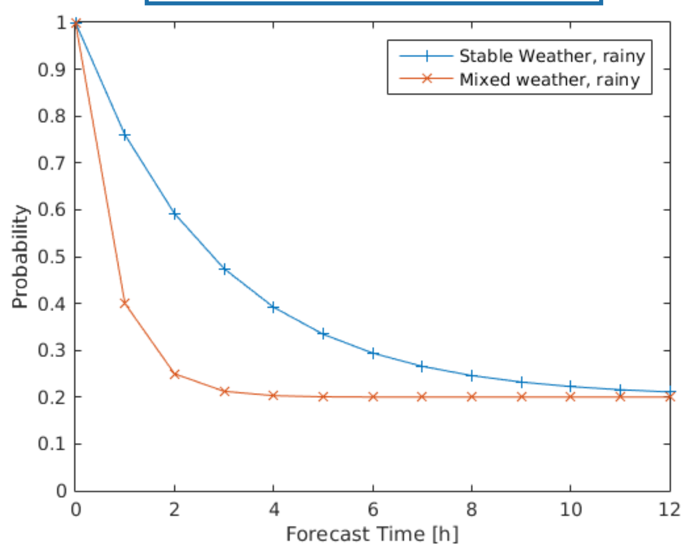
mixed $W_m = \begin{bmatrix} 0.85 & 0.6 \\ 0.15 & 0.4 \end{bmatrix}$

$\boldsymbol{p}^* = \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix}$ ← Long-term average probability of rain: **20%** → $\boldsymbol{p}^* = \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix}$

Eigenvalues: 1, **0.75**

Rainy now, next hours?

$RRRRDDDDDDDDDDDD$
$DDDDDDDDDDDDD\ldots$



Eigenvalues: 1, **0.25**

Rainy now, next hours?

$RDDDDDDDDDDDDDDDD$
$RDDDRDDDDDDDD\ldots$

Markov Chain: First Results

- *Aperiodic* and *irreducible* chains are *ergodic*:
  (every state reachable after $> N$ steps, irregular return time)
  - Convergence towards a unique *equilibrium distribution* $\boldsymbol{p}^*$
- Equilibrium distribution $\boldsymbol{p}^*$
  - Eigenvector of T with eigenvalue $\lambda = 1$:
  $$T\boldsymbol{p}^* = \boldsymbol{p}^*$$
  - Rate of convergence:
  Exponential decay with second largest eigenvalue $\propto |\lambda_2|^n$
- How to design a chain with a ***given*** equilibrium distribution?

Detailed Balance

- Detailed Balance is a local equilibrium

  Distribution p satisfies *detailed balance* if the total flow of probability between every pair of states is equal, the chain is then *reversible*:

$$P(l \leftarrow m)p(m) = P(m \leftarrow l)p(l)$$

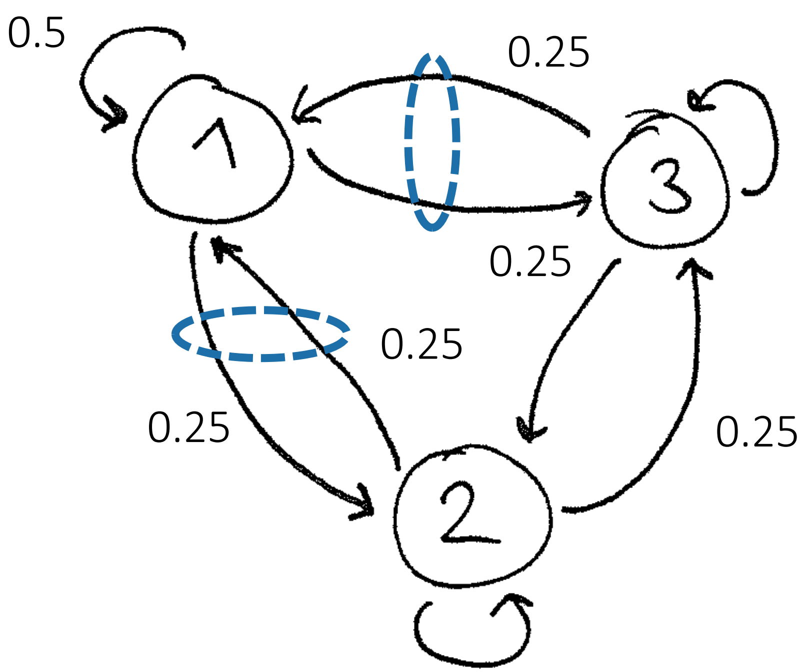
- Detailed balance implies: p is the equilibrium distribution

$$(T\boldsymbol{p})_l = \sum_m T_{lm} p_m = \sum_m T_{ml} p_l = p_l$$

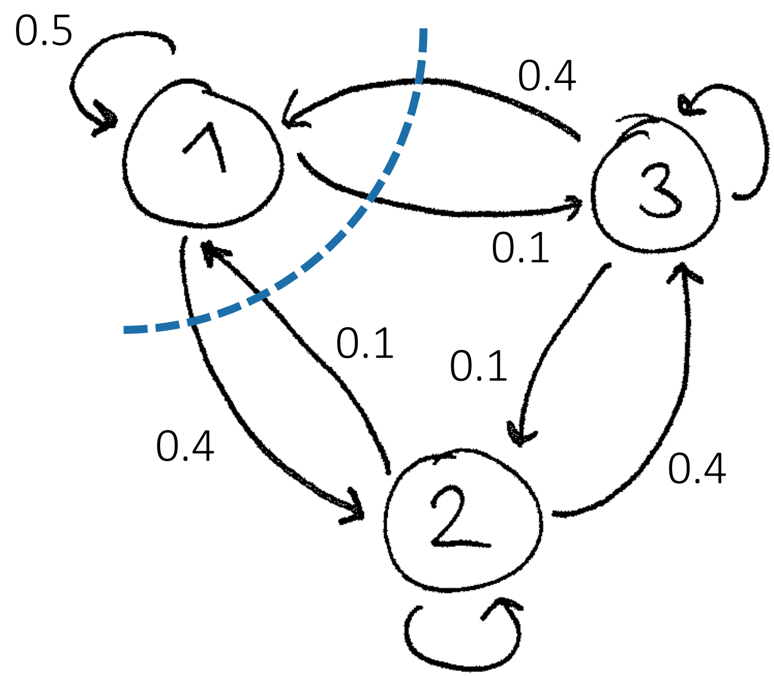
- Design Markov Chains with specific equilibrium distributions!

Example: Detailed Balance

- Local Equilibrium



- Global Equilibrium



- same equilibrium distribution [1/3, 1/3, 1/3]
- different convergence *mechanism*

Summary: Markov Chains

- Sequential random variables: $X_1, X_2, \dots$
- *Aperiodic* and *irreducible* chains are *ergodic*:
  - Convergence towards a unique *equilibrium distribution* $\boldsymbol{p}^*$
- Equilibrium distribution $\boldsymbol{p}^*$
  - Eigenvector of T with eigenvalue $\lambda = 1$: $T\boldsymbol{p}^* = \boldsymbol{p}^*$
  - Rate of convergence: decay with second largest eigenvalue $\propto |\lambda_2|^n$
- Detailed Balance:
  - Local equilibrium ⇒ global equilibrium
  - Easier to design Markov chains with given equilibrium distribution

The Metropolis Algorithm

MCMC to draw samples from an arbitrary distribution

The Metropolis Algorithm

Requirements:

- Proposal distribution $Q(\boldsymbol{x}'|\boldsymbol{x})$ – *must generate samples, symmetric*
- Target distribution $P(\boldsymbol{x})$ – *with point-wise evaluation*

Result:

- Stream of samples approximately from $P(\boldsymbol{x})$

- Initialize with sample $\boldsymbol{x}$
- Generate next sample, with current sample $\boldsymbol{x}$
  1. Draw a sample $\boldsymbol{x}'$ from $Q(\boldsymbol{x}'|\boldsymbol{x})$ ("proposal")
  2. With *probability* $\alpha = \min\left\{\frac{P(\boldsymbol{x}')}{P(\boldsymbol{x})}, 1\right\}$ accept $\boldsymbol{x}'$ as new state $\boldsymbol{x}$
  3. Emit current state $\boldsymbol{x}$ as sample

Properties

- **Approximation:** Samples $x_1, x_2, \ldots$ approximate $P(x)$

  Unbiased but correlated (not *iid*)

- **Normalization:** $P(x)$ does not need to be normalized

  Algorithm only considers ratios $P(x')/P(x)$

- **Dependent Proposals:** $Q(x'|x)$ depends on current sample x

  Algorithm adapts to target with simple 1-step memory

- **Symmetric Proposals:** $Q(x'|x) = Q(x|x')$

  Requirement of *Metropolis* algorithm

  Typical choice: Gaussian random walk $\mathcal{N}(x'|x, \sigma^2)$

Example: 2D Gaussian

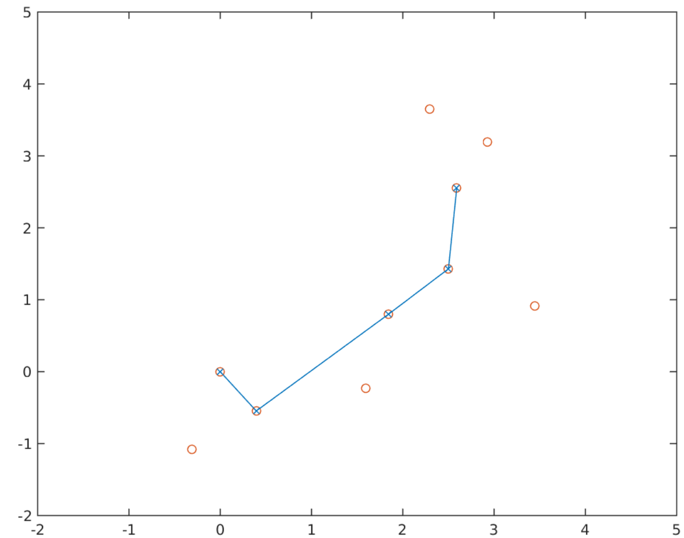
- Target: $P(\boldsymbol{x}) = \frac{1}{2\pi\sqrt{|\Sigma|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$
- Proposal: $Q(\boldsymbol{x}'|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}'|\boldsymbol{x}, \sigma^2 I_2)$ ← Random walk

Target

$$\mu = \begin{bmatrix} 1.5 \\ 1.5 \end{bmatrix}$$
$$\Sigma = \begin{bmatrix} 1.25 & 0.75 \\ 0.75 & 1.25 \end{bmatrix}$$

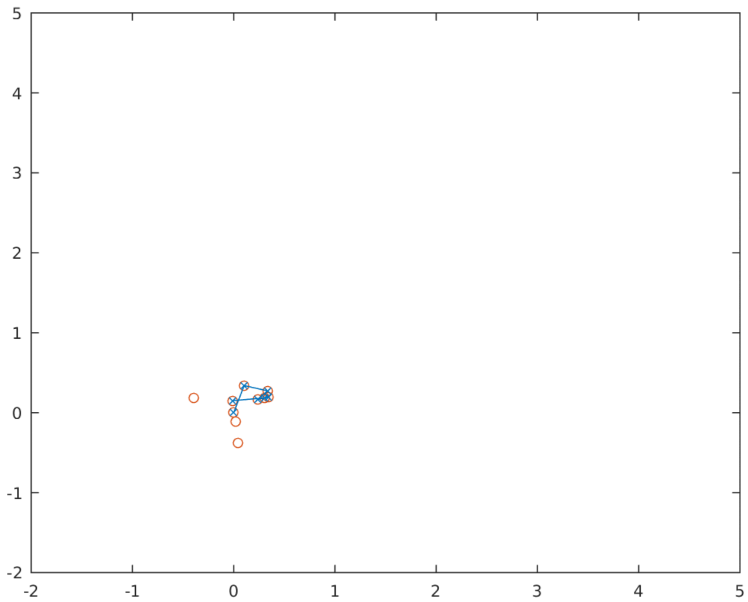
Sampled Estimate

$$\hat{\mu} = \begin{bmatrix} 1.56 \\ 1.68 \end{bmatrix}$$
$$\hat{\Sigma} = \begin{bmatrix} 1.09 & 0.63 \\ 0.63 & 1.07 \end{bmatrix}$$

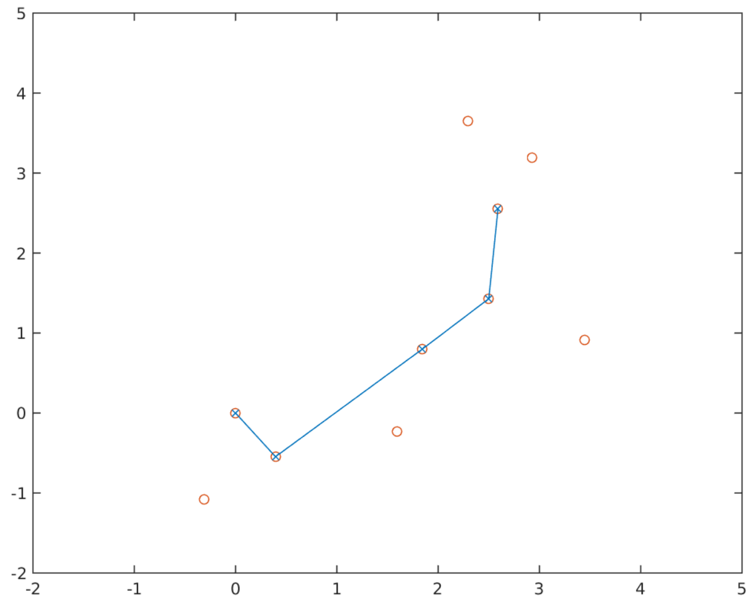


2D Gaussian: Different Proposals

$\sigma = 0.2$



$\sigma = 1.0$



Metropolis Algorithm: MCMC

- *Metropolis* defines a Markov chain with *equilibrium* distribution P

$$T_M(x' \leftarrow x) = Q(x'|x)\alpha(x'|x) + \sum_{\tilde{x}} Q(\tilde{x} \mid x)\big(1 - \alpha(\tilde{x}|x)\big)\, \delta_{x'x}$$

- Check: does detailed balance hold for P?

$$T_M(x' \leftarrow x)P(x) = T_M(x \leftarrow x')P(x')$$

- Expand: *(blackboard)*
- Result: P satisfies detailed balance for Metropolis kernel T_M
  - P is the stable distribution
  - P is the equilibrium distribution if the chain is irreducible
  - Samples from chain converge to be drawn from P!

Metropolis-Hastings Algorithm

- Extension to asymmetric Proposal distribution

$$Q(\boldsymbol{x}'|\boldsymbol{x}) \neq Q(\boldsymbol{x}|\boldsymbol{x}')$$

$$Q(\boldsymbol{x}'|\boldsymbol{x}) > 0 \Leftrightarrow Q(\boldsymbol{x}|\boldsymbol{x}') > 0$$

- Correction in acceptance probability

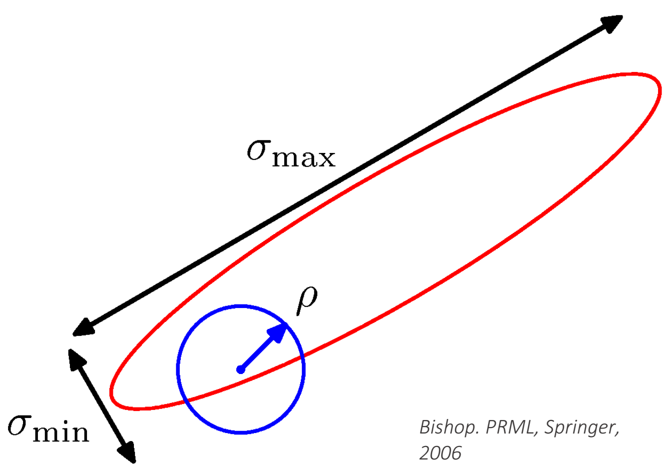
$$\alpha = \min\left\{\frac{P(x')}{P(x)}\frac{Q(x|x')}{Q(x'|x)}, 1\right\}$$

- Initialize with sample $\boldsymbol{x}$
- Generate next sample, with current sample $\boldsymbol{x}$
  1. Draw a sample $\boldsymbol{x}'$ from $Q(\boldsymbol{x}'|\boldsymbol{x})$ ("proposal")
  2. With *probability* $\alpha = \min\left\{\frac{P(x')}{P(x)}\frac{Q(x|x')}{Q(x'|x)}, 1\right\}$ accept $\boldsymbol{x}'$ as new state $\boldsymbol{x}$
  3. Emit current state $\boldsymbol{x}$ as sample

Metropolis: Limitations

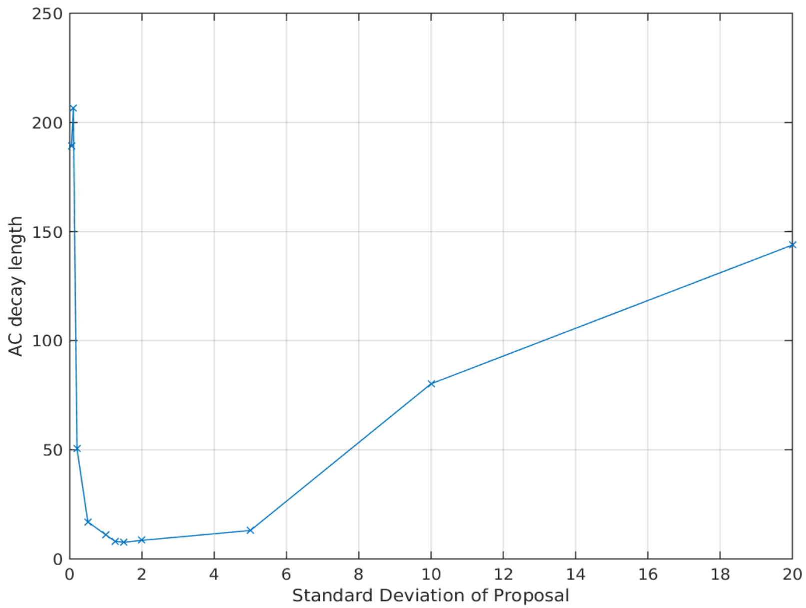
- Highly correlated targets

  Proposal should match target to avoid too many rejections



Bishop. PRML, Springer, 2006

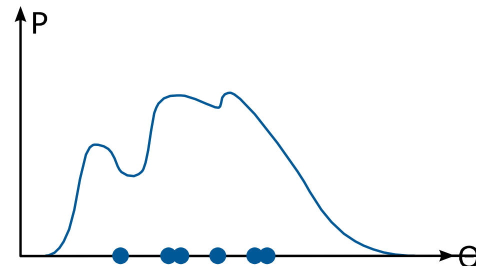
- Serial correlation
  - Results from rejection and too small stepping
  - Subsampling



Probabilistic Fitting with MCMC

- Probabilistic Registration
- Bayesian Inference
  - Posterior distribution
- Approximate Inference
- Sampling
  - *Simulate* posterior distribution
- Metropolis-Hastings
  - MCMC, general sampler
  - Sample from Q transform to P
  - Choose $P \propto p(\theta|I_R, I_T)$

$$p(\theta|I_R, I_T) = \frac{\ell(\theta; D)p(\theta)}{\int \ell(\theta; D)p(\theta)\mathrm{d}\theta}$$



$$\alpha = \min\left\{\frac{P(x')}{P(x)}\frac{Q(x|x')}{Q(x'|x)}, 1\right\}$$

Propose-and-Verify Algorithm

- Metropolis algorithm formalizes: *propose-and-verify*

Draw a sample x' from $Q(x'\|x)$	Propose
With *probability* $\alpha = \min\left\{\frac{P(\boldsymbol{x}')}{P(\boldsymbol{x})}, 1\right\}$ accept $\boldsymbol{x}'$ as new sample	Verify

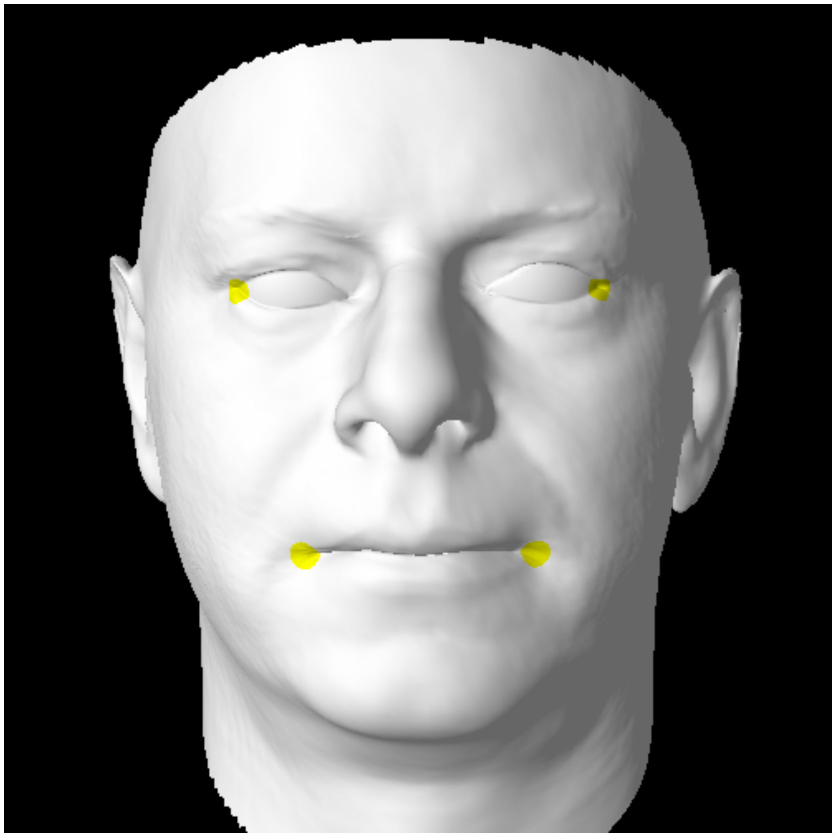
- Very useful concept to integrate unreliable proposals!
  - Can deal with heuristics which are not always right
  - Can deal with unreliable data
- Algorithmic advantage beyond probabilistic Bayesian concept

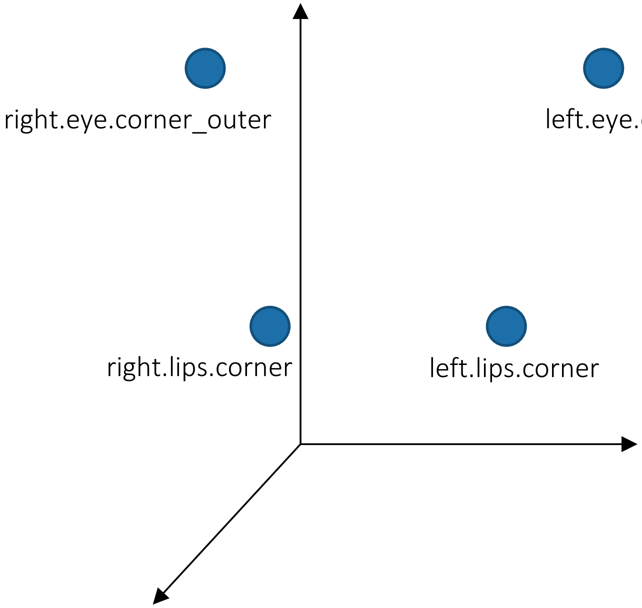
  Outlook: Filtering for 2D image analysis with unreliable data

Fitting 3D Landmarks

3D Alignment with Shape and Pose

3D Fitting Example





3D Fitting Setup

- 3D face with statistical model

  Discrete low-rank Gaussian Process

- Arbitrary rigid transformation

  Pose, Positioning in space

- Observations
  - Observed positions $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2 \dots, \tilde{\boldsymbol{x}}_\mathrm{L}$
  - Correspondence: $\boldsymbol{x}_1^\mathrm{r}, \boldsymbol{x}_2^\mathrm{r}, \dots, \boldsymbol{x}_\mathrm{L}^\mathrm{r}$

- Goal: Find Posterior Distribution

$$P(\theta|\tilde{\boldsymbol{x}}_1, \dots, \tilde{\boldsymbol{x}}_\mathrm{L}) \propto \ell(\tilde{\boldsymbol{x}}_1, \dots, \tilde{\boldsymbol{x}}_\mathrm{L}|\theta)P(\boldsymbol{\alpha})$$

- Parameters

$$\theta = (\boldsymbol{\alpha}, \varphi, \psi, \vartheta, \boldsymbol{t})$$

- Shape

$$\boldsymbol{x}' = \mu(\boldsymbol{x}) + \sum_i^d \alpha_i \sqrt{\lambda_i} \Phi_i(\boldsymbol{x})$$

- Rigid Transform
  - 3 angles (pitch, yaw, roll) φ, ψ, ϑ
  - Translation $\boldsymbol{t}$

$$\boldsymbol{x}' = R_\vartheta R_\psi R_\varphi(\boldsymbol{x}) + \boldsymbol{t}$$

Proposals

- Choose simple Gaussian random walk proposals (Metropolis)

$$"Q(\theta'|\theta) = N(\theta'|\theta, \Sigma_\theta)"$$

  - Normal *perturbations* of current state

- Block-wise to account for different parameter types
  - Shape $N(\boldsymbol{\alpha}'|\boldsymbol{\alpha}, \sigma_S^2 E_S)$
  - Rotation $N(\varphi'|\varphi, \sigma_\varphi^2) + N(\psi'|\psi, \sigma_\psi^2) + N(\vartheta'|\vartheta, \sigma_\vartheta^2)$
  - Translation $N(\boldsymbol{t}'|\boldsymbol{t}, \sigma_t^2 E_3)$

E_d Identity matrix (I is image)

- Large mixture distributions as proposals

$$Q(\theta'|\theta) = \sum c_i Q_i(\theta'|\theta)$$

3DMM Landmarks Likelihood

Simple models: **Independent Gaussians**

Observation of L landmark locations $\tilde{\boldsymbol{x}}_i$ in image

- Single *landmark position* model:

$$\boldsymbol{x}'_i(\theta) = R_{\varphi,\psi,\vartheta}\left(h_\alpha\left(\boldsymbol{x}_i^{\mathrm{ref}}\right)\right) + \boldsymbol{t}$$

$$\ell_i(\theta; \tilde{\boldsymbol{x}}_i) = N\left(\tilde{\boldsymbol{x}}_i | \boldsymbol{x}'_i(\theta), \sigma_{\mathrm{LM}}^2\right)$$

$\ell(\theta; D) = p(D|\theta)$

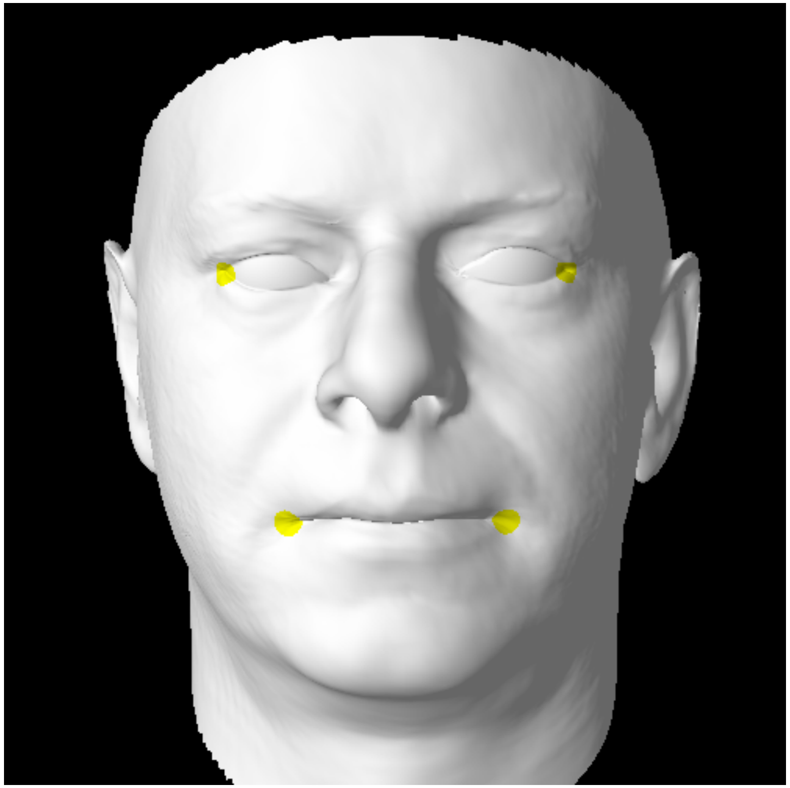
- *Independent* model (conditional independence):

$$\ell(\theta; \tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \dots, \tilde{\boldsymbol{x}}_L) = \prod_{i=1}^{L} \ell_i(\theta; \tilde{\boldsymbol{x}}_i)$$

- Independence and Gaussian are just *simple models* (questionable)

3D Fit to Landmarks

- Influence of landmarks uncertainty on final posterior?
    - $\sigma_{\mathrm{LM}} = 1\mathrm{mm}$
    - $\sigma_{\mathrm{LM}} = 4\mathrm{mm}$
    - $\sigma_{\mathrm{LM}} = 10\mathrm{mm}$
- Only 4 landmark observations:
    - Expect only weak shape impact
    - Should still constrain pose
- Uncertain LM should be looser



3D Fitting: Code

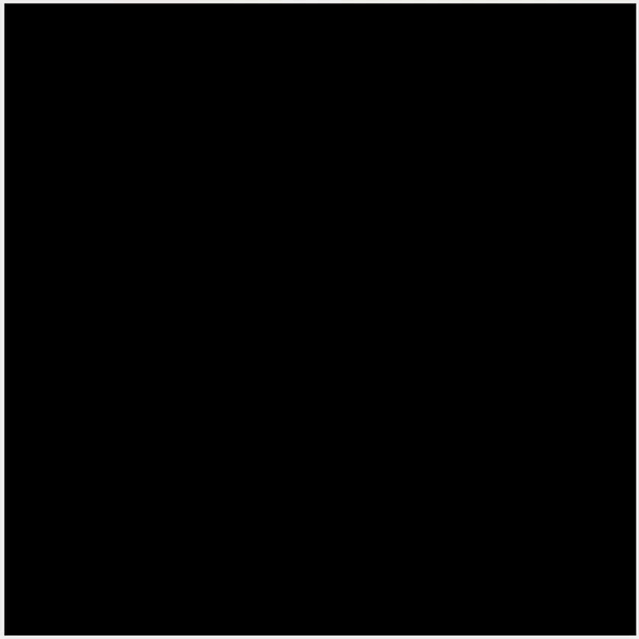
```
val yawProposal         = GaussianRotationProposal(AxisY, sdev = 0.05)
val pitchProposal       = GaussianRotationProposal(AxisX, sdev = 0.05)
val rollProposal        = GaussianRotationProposal(AxisZ, sdev = 0.05)
val rotationProposal    = MixtureProposal(
        0.6 *: yawProposal + 0.3 *: pitchProposal + 0.1 *: rollProposal)

val translationProposal = GaussianTranslationProposal(Vector(2, 2, 2))

val poseProposal          = MixtureProposal(
        rotationProposal + translationProposal)

val shapeProposal         = GaussianShapeProposal(sdev = 0.05)

val lmFitter = MetropolisHastings(
   proposal  = MixtureProposal(0.2 *: poseProposal +  0.8 *: shapeProposal),
   evaluator = ProductEvaluator(lmLikelihood * shapePrior))

val samples = lmFitter.iterator(initState).drop(2000).take(8000).toIndexedSeq
```

Posterior: Pose & Shape, 4mm

Target

Pose

Shape



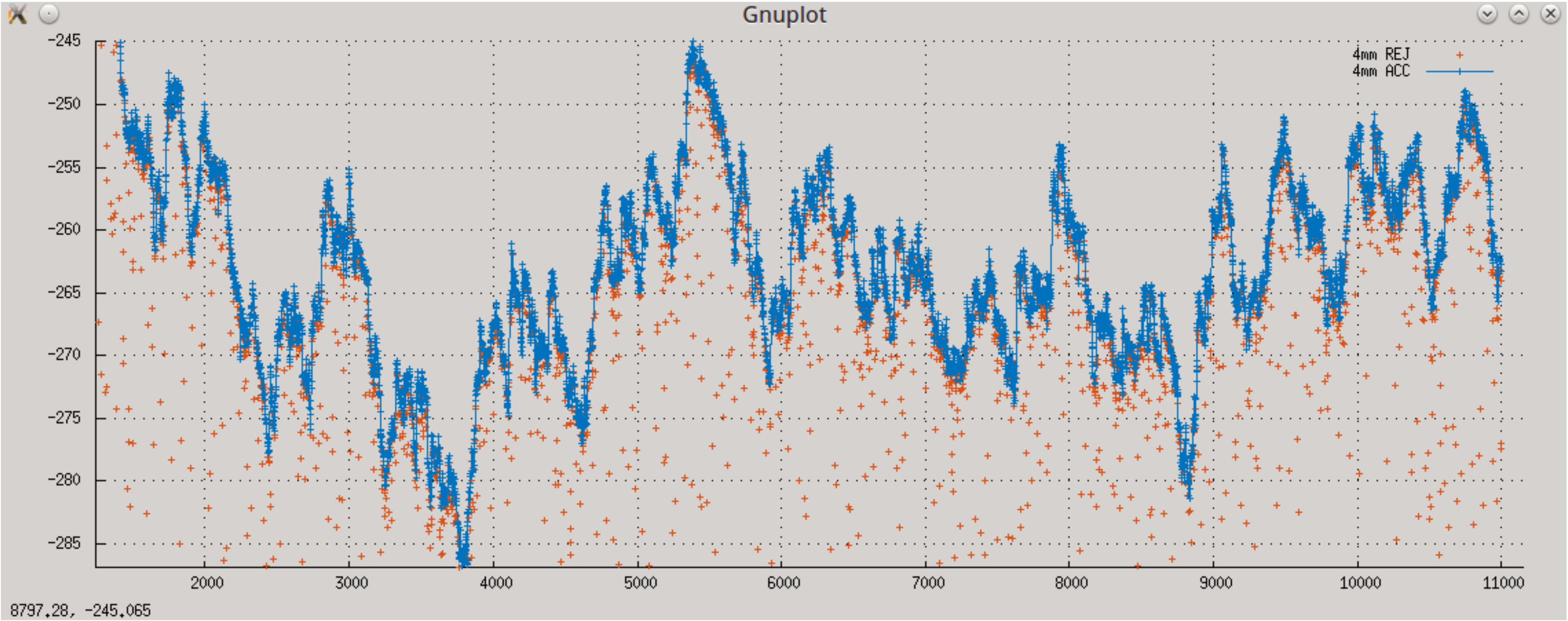
$\hat{\mu}_{\mathrm{yaw}} = 0.511$
$\hat{\sigma}_{\mathrm{yaw}} = 0.073\ (4°)$

$\hat{\mu}_{\mathrm{t_x}} = -1\ \mathrm{mm}$
$\hat{\sigma}_{\mathrm{t_x}} = 4\ \mathrm{mm}$

$\hat{\mu}_{\alpha_1} = 0.4$
$\hat{\sigma}_{\alpha_1} = 0.6$

(Estimation from samples)

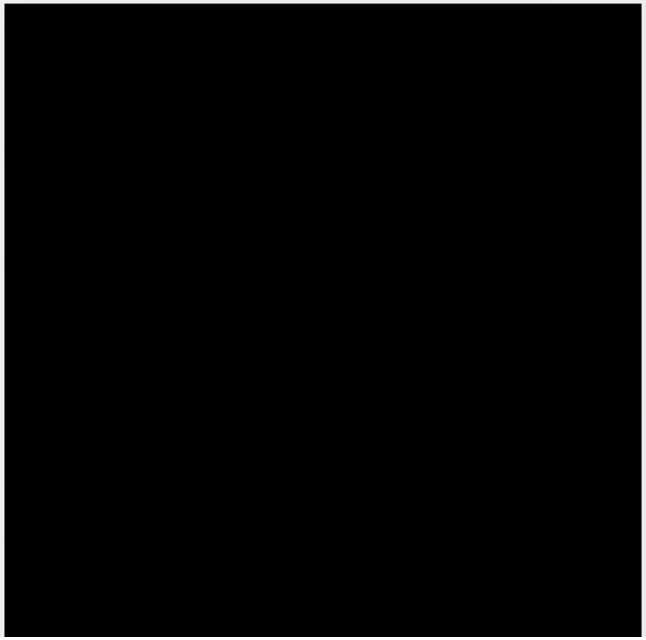
Posterior: Pose & Shape, 4mm



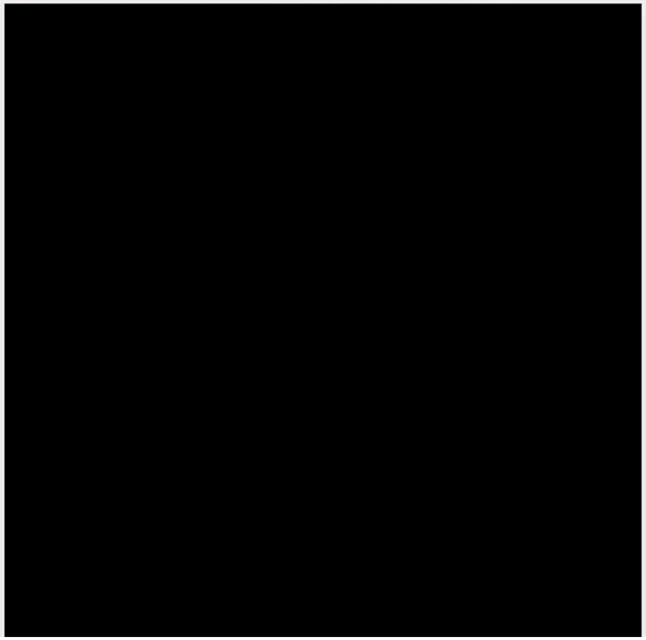
Posterior values (log, unnormalized!)

Posterior: Pose & Shape, 1mm

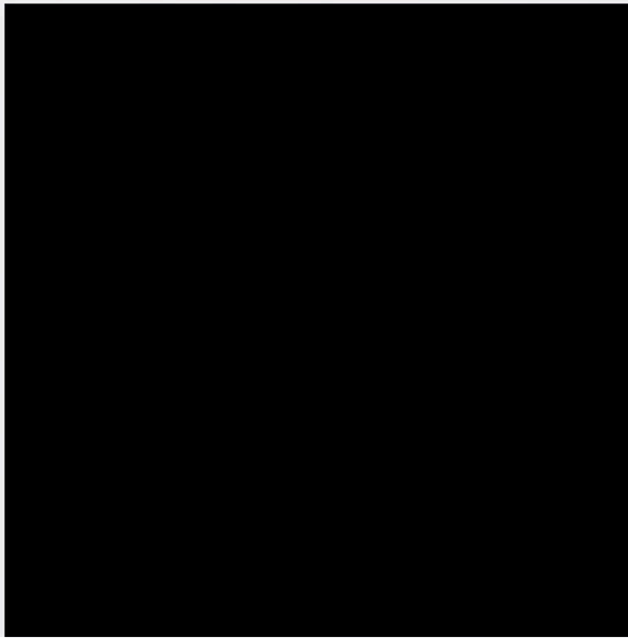
Target



Pose



Shape



$\hat{\mu}_{\mathrm{yaw}} = 0.50$
$\hat{\sigma}_{\mathrm{yaw}} = 0.041\ (2.4°)$

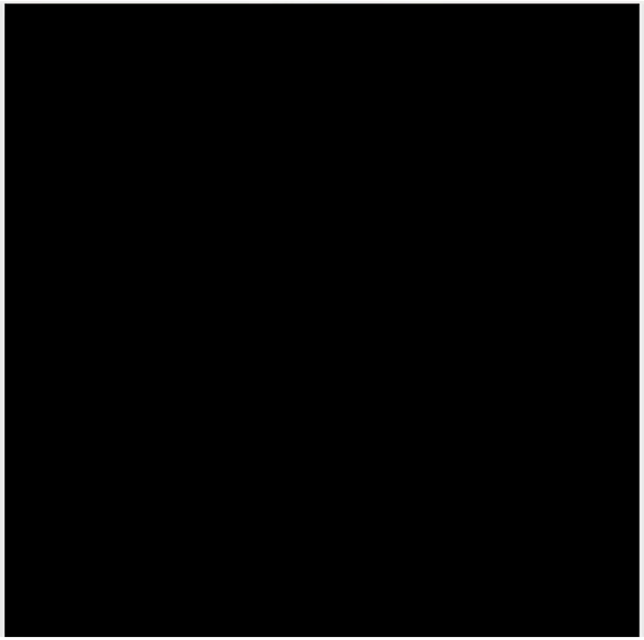
$\hat{\mu}_{\mathrm{t_x}} = -2\ \mathrm{mm}$
$\hat{\sigma}_{\mathrm{t_x}} = 0.8\ \mathrm{mm}$

$\hat{\mu}_{\alpha_1} = 1.5$
$\hat{\sigma}_{\alpha_1} = 0.35$

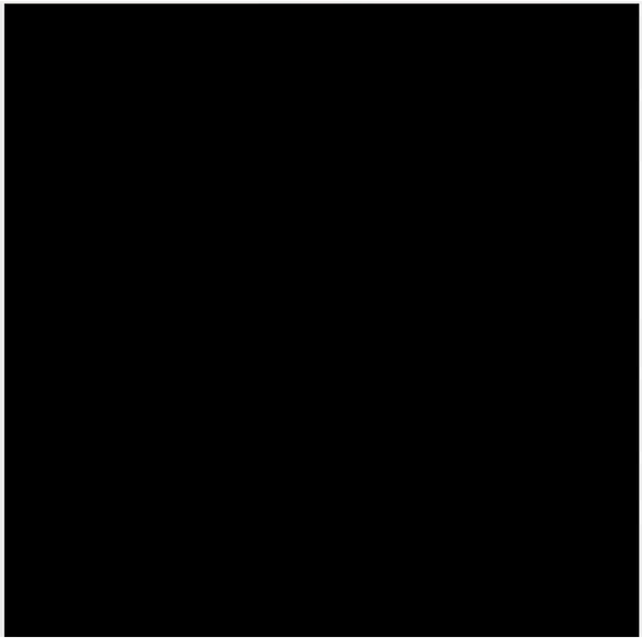
Posterior: Pose & Shape, 10mm

Target

Pose



Shape



$\hat{\mu}_{\text{yaw}} = 0.49$
$\hat{\sigma}_{\text{yaw}} = 0.11\ (7°)$

$\hat{\mu}_{\text{t}_\text{x}} = -5\ \text{mm}$
$\hat{\sigma}_{\text{t}_\text{x}} = 10\ \text{mm}$

$\hat{\mu}_{\alpha_1} = 0$
$\hat{\sigma}_{\alpha_1} = 0.6$

Summary: MCMC for 3D Fitting

- Probabilistic inference for fitting probabilistic models
  - Bayesian inference: posterior distribution
- Probabilistic inference is often intractable
  - Use *approximate* inference methods
- Sampling methods approximate by *simulation*
- MCMC methods provide a powerful sampling framework
  - Markov Chain with target distribution as equilibrium distribution
  - General algorithms, e.g. Metropolis-Hastings
- 3D landmarks fitting example: Posterior distribution
  - Model likelihood
  - Define proposals

Overview

- *Computer Graphics Overview*
- *Probabilistic Setup*
- *Markov Chain Monte Carlo*
  - *Markov Chains*
- *3D Fitting Problem*
  - *Landmarks*
- 2D Face Image Analysis
  - Image fitting
  - Filtering with unreliable information