1 System Modelling

Continuous time (linear):

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \boldsymbol{x}(0) = \boldsymbol{x}_0 \qquad \boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t)$$

Continuous time (nonlinear):

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \boldsymbol{x}(0) = \boldsymbol{x}_0 \qquad \boldsymbol{y}(t) = \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t))$$

Discrete time (linear):

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_d\boldsymbol{x}_k + \boldsymbol{B}_d\boldsymbol{u}_k \qquad \boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{D}\boldsymbol{u}_k$$

With a zero-order hold:

$$\begin{bmatrix} \boldsymbol{A}_d & \boldsymbol{B}_d \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} = \exp\left(\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} h\right) \qquad \boldsymbol{A}_d = e^{\boldsymbol{A}h}, \boldsymbol{B}_d = \int_0^h e^{\boldsymbol{A}\tau'}\mathrm{d}\tau'\boldsymbol{B}$$

2 Numerical Stability

Well conditioned problem: Cond_x small or $\mathrm{cond}_x \leq 1$

$$\mathrm{Cond}_x = \left|\frac{\Delta\bar{y}}{\Delta x}\right| \approx \left|\frac{\mathrm{d}f}{\mathrm{d}x}\right| \qquad \mathrm{cond}_x = \left|\frac{\delta\bar{y}}{\delta x}\right| \approx |K_x| = \left|\frac{\mathrm{d}f}{\mathrm{d}x}\cdot\frac{x}{f(x)}\right|$$

φ is *order* p or $O(\Delta^p)$ if: $\tilde{y} = \varphi(x, \Delta) - f(x) \propto \Delta^p$

Consistency: $\lim_{\Delta\to 0}\varphi(x, \Delta) = f(x)$ Stability: $\left|\frac{\Delta\tilde{y}}{\Delta x}\right| \approx \left|\frac{\mathrm{d}\varphi}{\mathrm{d}x}\right| < 1$

$\hat{y} = \varphi(\hat{x}, \Delta) = \varphi(\hat{x} + \Delta x)$ converges to $f(x)$ for $\Delta \to 0$ if it's consistent and at least marginally stable.

$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ well-conditioned/stable when $\mathrm{Re}(\lambda) < 0$ for all λ.

$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}$ well-conditioned/stable when $|\lambda| < 1$ for all λ.

3 Numerical Root Finding

f is *Lipschitz continuous* if

$$\exists c \text{ s.t. } \forall x, z \in \mathbb{R}, |f(x) - f(z)| \leq c|x - z|$$

An algorithm has *convergence rate/order* r if: $\lim_{k\to\infty}\frac{|E_{k+1}|}{|E_k|^r} = C$

Bisection: order 1; requires continuity; have l_k, r_k such that $\operatorname{sgn} f(l_k) \neq \operatorname{sgn} f(r_k)$, check $c = \frac{l_k + r_k}{2}$ each iteration.

Fixed-point Iteration: order 2 if $g'(x^*) \approx 0$, 1 otherwise; requires update function $g(x^*) = x^*$ for Lipschitz g with $c < 1$ for $x \approx x^*$. To find $g(x)$, rearrange $f(x^*) = 0$ into the form $x^* = g(x^*)$. Start with guess, update as $x_{k+1} = g(x_k)$.

Newton's Method: order 2; requires continuous differentiability; equivalent to fixed-point iteration with $g(x) = x - \frac{f(x)}{f'(x)}$. Start with guess, update as $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$. Exact for linear f and suffers for highly nonlinear f.

Secant Method: order 1.6; Newton with $f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$.

4 Numerical Integration and Differentiation

Midpoint/Trapezoidal Rule: order 2, approx. f as const./linear

$$\int_{x_i}^{x_{i+1}} f(x)\,\mathrm{d}x \approx (x_{i+1} - x_i)\cdot f\left((x_{i+1} + x_i)/2\right) \approx (x_{i+1} - x_i)\cdot\frac{f(x_{i+1}) + f(x_i)}{2}$$

Simpson's Rule: order 4, approx. f as quadratic

$$(x_{i+1} - x_i)\cdot\frac{f(x_{i+1}) + 4f\left((x_{i+1} + x_i)/2\right) + f(x_i)}{6}$$

Differentiation: Forward (order 1), Backward (1), Centered (2) Difference

$$f'(x) \approx \frac{f(x + \Delta) - f(\Delta)}{\Delta} \approx \frac{f(x) - f(x - \Delta)}{\Delta} \approx \frac{f(x + \Delta) - f(x - \Delta)}{2\Delta}$$

5 Numerical ODE Solving

Forward Euler's Method: order 1 (global); explicit; conditionally stable if $|1 + h\lambda| < 1$: $x_{k+1} = x_k + hf_k$

Backward Euler's Method: order 1 (global); implicit; conditionally stable if $|1 - h\lambda| > 1$: $x_{k+1} = x_k + hf_{k+1}$

Trapezoidal Method: order 2 (global); implicit; conditionally stable if $\left|\frac{1 + \frac{1}{2}h\lambda}{1 - \frac{1}{2}h\lambda}\right| < 1$: $x_{k+1} = x_k + \frac{h}{2}(f_{k+1} + f_k)$

Heun's Method (RK2): order 2 (global); explicit; conditionally stable if $-4 < 2h\lambda + h^2\lambda^2 < 0$: $x_{k+1} = x_k + \frac{h}{2}\left(f(x_k + hf_k) + f_k\right)$

Fourth-Order Runge-Kutta (RK4): order 4 (global); explicit.

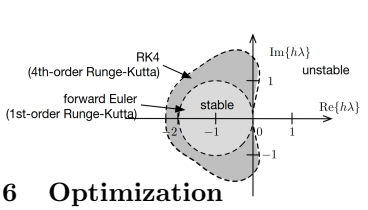
$$x_{k+1} = x_k + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

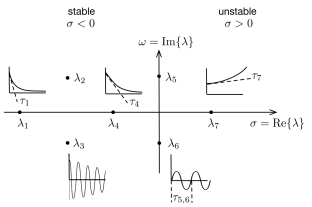
$k_1 = f(x_k), \quad k_2 = f\left(x_k + \frac{1}{2}hk_1\right) \quad k_3 = f\left(x_k + \frac{1}{2}hk_2\right) \quad k_4 = f(x_k + hk_3)$

Time constant: for $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ with complex eigenvalues λ_i:

$$\tau_i = \left\{\frac{1}{|\mathrm{Re}(\lambda_i)|}, \frac{2\pi}{|\mathrm{Im}(\lambda_i)|}\right\} \qquad \gamma = \frac{\tau_{\max}}{\tau_{\min}} \text{ stiff if } > 10^3$$

Rule of thumb: simulate $T = 5\tau_{\max}$ if system is stable; use step size $h = \min\left\{\frac{\tau_{\min}}{10}, \frac{T}{200}\right\}$ or number of steps $k = \max\{50\gamma, 200\}$; plot with step size $H = \frac{T}{200} = \frac{\tau_{\max}}{40}$; for stiff systems use a variable-step solver with initial step $\frac{\tau_{\min}}{10}$. Use adaptive step size solvers for stiff problems.





6 Optimization

Formulation: minimize $f(\boldsymbol{x})$, subject to $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}$ (equality constraints) and $\boldsymbol{h}(\boldsymbol{x}) \geq \boldsymbol{0}$ (inequality constraints); $f : \mathbb{R}^n \mapsto \mathbb{R}, \boldsymbol{g} : \mathbb{R}^n \mapsto \mathbb{R}^m, \boldsymbol{h} : \mathbb{R}^n \mapsto \mathbb{R}^p$.

$\boldsymbol{x}^*$ is a *global minimum* if $\forall \boldsymbol{x} \in \mathbb{R}^n, f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$;

or *local minimum* if $\exists \varepsilon > 0$ s.t. $\forall \boldsymbol{x} \in \mathbb{R}^n, \|\boldsymbol{x} - \boldsymbol{x}^*\|_2 < \varepsilon \implies f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$

All minima of f are *stationary points*, where $\vec{\nabla} f(\boldsymbol{x}) = \boldsymbol{0}$; to find the global minimum, check all stationary points and boundaries.

$$\boldsymbol{H}_f(\boldsymbol{x}) = \underset{i,j}{\text{matrix}}\left[\frac{\partial^2 f}{\partial x_i \partial x_j}\right]$$

Local minimum if $\boldsymbol{H}_f(\boldsymbol{x}^*) > 0$, maximum if $\boldsymbol{H}_f < 0$, saddle if $\boldsymbol{H}_f = 0$.

A *feasible point* is any $\boldsymbol{x}$ satisfying all constraints; the *feasible set* contains all feasible points. A *critical point* is a local maximum, minimum or saddle point in the feasible set.

Lagrange Multipliers: for equality constraints $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}$:

$$\min_{\boldsymbol{x},\boldsymbol{\lambda}} \Lambda(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) - \boldsymbol{\lambda}^T\boldsymbol{g}(\boldsymbol{x})$$

Karush-Kuhn-Tucker (KTT) Conditions: $\boldsymbol{x}^*$ is a critical point when there exists $\boldsymbol{\lambda} \in \mathbb{R}^m$ and $\boldsymbol{\mu} \in \mathbb{R}^p$ such that:

1. Stationarity: $\vec{\nabla} f(\boldsymbol{x}^*) - \sum_i \lambda_i \vec{\nabla} g_i(\boldsymbol{x}^*) - \sum_j \mu_j \vec{\nabla} h_j(\boldsymbol{x}^*) = \boldsymbol{0}$
2. Primal feasibility: $\boldsymbol{g}(\boldsymbol{x}^*) = \boldsymbol{0}$ and $\boldsymbol{h}(\boldsymbol{x}^*) \geq \boldsymbol{0}$
3. Complementary slackness: $\forall j, \mu_j h_j(\boldsymbol{x}^*) = 0$
4. Dual feasibility: $\forall j, \mu_j \geq 0$

f is *convex* if $\boldsymbol{H}_f > 0$ for all $\boldsymbol{x}$, or

$$\forall \boldsymbol{x}_1 \neq \boldsymbol{x}_2, \forall \alpha \in (0, 1), f((1 - \alpha)\boldsymbol{x}_1 + \alpha\boldsymbol{x}_2) \leq (1 - \alpha)f(\boldsymbol{x}_1) + \alpha f(\boldsymbol{x}_2)$$

A set S is *convex* if: $\forall \boldsymbol{x}, \boldsymbol{y} \in S, \alpha \in [0, 1], \alpha\boldsymbol{x} + (1 - \alpha)\boldsymbol{y} \in S$

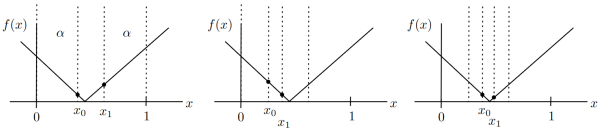
or $S = \{\, \boldsymbol{x} \mid f(\boldsymbol{x}) \leq c \,\}$ for some convex $f(\boldsymbol{x})$.

If the objective f and feasible set are both convex, the problem is convex and has a unique minimum.

7 Numerical Optimization Algorithms

f is *unimodular* over $[a, b]$ if there exists $x^* \in [a, b]$ s.t. f is decreasing over $[a, x^*]$ and increasing over $[x^*, b]$.

Golden Section Search: 1D, unconstrained; order 1; requires unimodularity. Rescale to $[0, 1]$, choose $x_0 = \alpha, x_1 = 1 - \alpha$ for $0 < \alpha < 1/2$; evaluate $f(x_0), f(x_1)$, discard larger side, rescale and repeat. Using $1 - \alpha = \frac{1}{2}(\sqrt{5} - 1)$ allows using x_0 as the next x_1.



Gradient Descent: unconstrained; requires twice differentiable. Let $g(\alpha) = f(\boldsymbol{x}_k - \alpha\vec{\nabla} f(\boldsymbol{x}_k)^T)$, find α^* minimizing g, update $x_{k+1} = x_k - a^*\vec{\nabla} f(\boldsymbol{x}_k)^T$. Optimizing g is expensive so fixed step size can be used. Suffers from poor conditioning of f.

Newton's Method: unconstrained; requires twice differentiable. Iterate as $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{H}_f^{-1}(\boldsymbol{x}_k)\vec{\nabla} f(\boldsymbol{x}_k)^T$; or in 1D, $x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$. Gauss-Newton approximates $\boldsymbol{H}_f$ using first derivatives (mix of gradient descent and Newton). Levenberg-Marquardt applies adaptive regularization for nearly-singular Hessians.

Sequential Quadratic Programming (SQP): constrained. Iteratively solves a series of simpler, less constrained approximations of the problem. f is replaced by a quadratic approximation and the constraints linearized. Similar to Newton's method; only converges if initial guess is good.

Barrier Methods: constrained. Constraints are turned into penalties on the objective. Define new objective as $f'(x) = f(x) + \rho\frac{1}{h(x)}$ where weight ρ is increased to satisfy constraints, decreased for more accuracy.

8 Linear Algebra

$\boldsymbol{A}^T\boldsymbol{A}$ is positive definite if $\boldsymbol{A}$ is full rank, semi-definite otherwise.

Orthogonal $\boldsymbol{Q}$ rotates, preserves angles, and $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{1}$.

$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ can be solved by factoring $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$ and solving $\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b}, \boldsymbol{U}\boldsymbol{x} = \boldsymbol{y}$.

A general *vector norm* satisfies: $\|\boldsymbol{x}\| = 0 \iff \boldsymbol{x} = \boldsymbol{0}$; $\|c\boldsymbol{x}\| = |c|\|\boldsymbol{x}\|$; $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$. The *p-norm* for $p \geq 1$ is convex:

$$\|\boldsymbol{x}\|_p = \left(|x_1|^p + |x_2|^p + \cdots + |x_n|^p\right)^{\frac{1}{p}}$$

2-norm is Euclidean; ∞-norm is $\|\boldsymbol{x}\|_\infty = \max(|x_1|, |x_2|, \ldots, |x_n|)$.

The *matrix norm* induced by a vector norm is

$$\|\boldsymbol{A}\| = \max\{\, \|\boldsymbol{A}\boldsymbol{x}\| \mid \|\boldsymbol{x}\| = 1 \,\} = \max_{\boldsymbol{x}\in\mathbb{R}^n, \boldsymbol{x}\neq 0} \frac{\|\boldsymbol{A}\boldsymbol{x}\|}{\|\boldsymbol{x}\|}$$

1-norm: max col. sum; *2-norm/spectral radius:* "largest eigenvalue" of $\boldsymbol{A}$

$$\|\boldsymbol{A}\|_1 = \max_{i\le j\le n} \sum_{i=1}^{m} |a_{ij}| \qquad \|\boldsymbol{A}\|_2 = \max\left\{ \sqrt{\lambda} \;\middle|\; \exists \boldsymbol{x} \in \mathbb{R}^n \text{ s.t. } \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x} \right\}$$

Frobenius norm: always $\|\boldsymbol{A}\|_2 \le \|\boldsymbol{A}\|_F$; ∞*-norm:* maximum row sum

$$\|\boldsymbol{A}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{\operatorname{tr} \boldsymbol{A}^T\boldsymbol{A}} \qquad \|\boldsymbol{A}\|_\infty = \max_{1\le i\le m} \sum_{j=1}^{n} |a_{ij}|$$

The *condition number* of $\boldsymbol{A}$ with respect to a norm is $\operatorname{cond} \boldsymbol{A} = \|\boldsymbol{A}\|\|\boldsymbol{A}^{-1}\|$ or ∞ if $\boldsymbol{A}$ is non-invertible. For solving $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, this describes how error in $\boldsymbol{A}$ and $\boldsymbol{b}$ propagates to $\boldsymbol{x}$.

Eigendecomposition: $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}$ where $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{\lambda}\boldsymbol{v}$.

Singular Value Decomposition: $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ where $\boldsymbol{V} \in \mathbb{R}^{n\times n}, \boldsymbol{U} \in \mathbb{R}^{m\times m}$ are orthogonal, $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times n}$ is rectangular diagonal with singular values. Singular values and vectors are related by $\boldsymbol{A}\boldsymbol{v}_i = \sigma_i \boldsymbol{u}_i$.
$\sigma_i = \sqrt{\lambda_i(\boldsymbol{A}^T\boldsymbol{A})} = \sqrt{\lambda_i(\boldsymbol{A}\boldsymbol{A}^T)}$, total $\max\{m, n\}$ singular values, nonnegative and real, in decreasing order with $k = \operatorname{rank} \boldsymbol{A}$ nonzero singular values.
$\boldsymbol{V}$ has k eigenvectors of $\boldsymbol{A}^T\boldsymbol{A}$ and $n-k$ orthonormal vectors from $\ker(\boldsymbol{A}^T\boldsymbol{A})$;
$\boldsymbol{U}$ has k eigenvectors of $\boldsymbol{A}\boldsymbol{A}^T$ and $m-k$ orthonormal vectors from $\ker(\boldsymbol{A}\boldsymbol{A}^T)$.
The rank $j \le k$ approximation of $\boldsymbol{A}$ is $\sum_{i=1}^{j} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$.

9 Probability Definitions and Theorems

The *cumulative distribution function* (CDF) of $f_x(x)$ is

$$F_x(x) = \sum_{\bar{x}=-\infty}^{x} f_x(x)$$

Given RVs $x \in \mathcal{X}, y \in \mathcal{Y}$, *marginalization* (sum rule) and *conditioning* (product rule) are given by:

$$f_x(x) = \sum_{y\in\mathcal{Y}} f_{xy}(x,y) \qquad f_{x|y}(x|y) = \frac{f_{xy}(x,y)}{f_y(y)}$$

Which also apply to conditional PDFs; given $z \in \mathcal{Z}$:

$$f_{x|z}(x|z) = \sum_{y\in\mathcal{Y}} f_{xy|z}(x,y|z) \qquad f_{x|y,z}(x|y,z) = \frac{f_{xy|z}(x,y|z)}{f_{y|z}(y|z)}$$

The above leads to *total probability theorem* and *Bayes' theorem:*

$$f_x(x) = \sum_{y\in\mathcal{Y}} f_{x|y}(x|y) f_y(y) \qquad f_{x|y}(x|y) = \frac{f_{y|x}(y|x) f_x(x)}{f_y(y)}$$

x and y are *independent* if $f(x|y) = f(x) \iff f(x,y) = f(x)f(y)$.
x and y are *conditionally independent* given z if

$$f(x|y,z) = f(x|z) \iff f(x,y|z) = f(x|z)f(y|z)$$

The *expected value* or *mean* of $\boldsymbol{x}$ is $\boldsymbol{\mu}_x = \mathbb{E}[\boldsymbol{x}] = \sum_{\boldsymbol{x}\in\mathcal{X}} \boldsymbol{x} f_x(\boldsymbol{x})$

The *(co)variance* of $\boldsymbol{x}$ is

$$\boldsymbol{\Sigma}_x = \operatorname{Var}[\boldsymbol{x}] = \mathbb{E}[(\boldsymbol{x} - \mathbb{E}[\boldsymbol{x}])(\boldsymbol{x} - \mathbb{E}[\boldsymbol{x}])^T] = \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^T] - \mathbb{E}[\boldsymbol{x}]\mathbb{E}[\boldsymbol{x}]^T$$

$\operatorname{Var}[x] = \sigma_x^2 = \mathbb{E}[x^2] - (\mathbb{E}[x])^2$ in 1D. $\boldsymbol{\Sigma}$ is symmetric positive (semi-)definite.
Law of the Unconscious Statistician:

$$\mathcal{Y} = \{\, y \mid y = g(x), x \in \mathcal{X} \,\} \implies \mathbb{E}[y] = \mathbb{E}[g(x)]$$

Change of Variables: Given $f_y(y)$, let $x = g(y)$; then

$$\text{Discrete: } f_x(x_j) = \sum_{y_{j,i}\in\mathcal{Y}_j} f_y(y_{j,i}) \qquad \text{Continuous: } f_x(x) = \frac{1}{\frac{\mathrm{d}g(y)}{\mathrm{d}y}} f_y(y)$$

where $\mathcal{Y}_j$ is the set of all $y_{j,i}$ such that $g(y_{j,i}) = x_j$ (discrete); and $g(y)$ is required to be continuously differentiable and strictly increasing/decreasing (continuous only).

10 Bayesian Tracking

Formulation: computing $f(\boldsymbol{x}_k|\boldsymbol{y}_{1:k})$ from $f(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1})$, given some motion model $\boldsymbol{x}_k = \boldsymbol{f}_{k-1}(\boldsymbol{x}_{k-1}, \boldsymbol{v}_{k-1})$ and observation model $\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{x}_k, \boldsymbol{w}_k)$, where $\boldsymbol{x}_k, \boldsymbol{w}_k$ are independent with known PDFs, and assuming the *Markov property:* $f(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{x}_{k-2}, \dots) = f(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$.

Prior Update: $f(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}) = \sum_{\boldsymbol{x}_{k-1}\in\mathcal{X}} f(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) f(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1})$

$f(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ can be computed from motion model and noise distribution by change of variables.

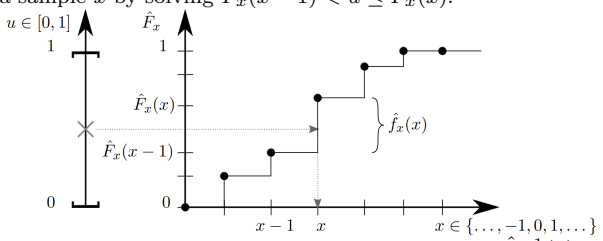
Measurement Update: $f(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}) = \dfrac{f(\boldsymbol{y}_k|\boldsymbol{x}_k) f(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1})}{\sum_{\boldsymbol{\chi}_k\in\mathcal{X}} f(\boldsymbol{y}_k|\boldsymbol{\chi}_k) f(\boldsymbol{\chi}_k|\boldsymbol{y}_{1:k-1})}$

$f(\boldsymbol{y}_k|\boldsymbol{x}_k)$ can be computed from measurement model and noise distribution by change of variables. The term in the denominator is for normalization.

11 Sampling From Distributions

Given u sampled from $f_u(u)$, a continuous uniform distribution over $(0, 1)$:

One Variable, Discrete: Given a desired PDF $\hat{f}_x(x)$, remap so $\mathcal{X} = \mathbb{Z}$, obtain a sample x by solving $\hat{F}_x(x-1) < u \le \hat{F}_x(x)$.



One Variable, Continuous: Obtain sample by $x = \hat{F}^{-1}(u)$ if CDF is invertible; otherwise pick any x value satisfying $\hat{F}(x) = u$.

Multiple Variables (Finite Sample Space): Given a desired discrete joint PDF $\hat{f}_{xy}(x,y)$, let $\mathcal{Z} = 1, 2, \dots, N$ where N is the number of unique combinations of (x,y); define $\hat{f}_z(z) = \hat{f}_{xy}(x,y)$ for some one-to-one mapping $z \leftrightarrow (x,y)$; sample z using the standard method, then use the mapping to convert to (x,y).

Multiple Variables (General): Find $\hat{f}_y(y)$ and $\hat{f}_{xy}(x,y) = \hat{f}_{x|y}(x|y)\hat{f}(y)$; sample $\hat{f}_y(y)$ for y, substitute into $\hat{f}_{x|y}(x|y)$ to sample for x.

12 Extracting Estimates from PDFs

Goal: Given observation $\boldsymbol{y}$ and observation model $f(\boldsymbol{y}|\boldsymbol{x})$, estimate $\boldsymbol{x}$.

Maximum Likelihood (ML): $\hat{\boldsymbol{x}}^{\mathrm{ML}} = \arg\max_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{y}|\boldsymbol{x})$

Finds peak of distribution; best used when $f(\boldsymbol{x})$ is not known. More sensitive to outliers and modelling error, since the distribution peak might not be the most robust option.

Maximum *a Posteriori* (MAP): $\hat{\boldsymbol{x}}^{\mathrm{MAP}} = \arg\max_{\boldsymbol{x}} f(\boldsymbol{y}|\boldsymbol{x}) f(\boldsymbol{x})$

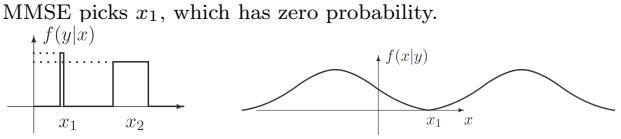
Finds peak of distribution $f(\boldsymbol{x}|\boldsymbol{y})$ given known distribution of $f(\boldsymbol{x})$. Suffers from the same sensitivity issues of ML.

Minimum Mean Squared Error (MMSE):

$$\hat{\boldsymbol{x}}^{\mathrm{MMSE}} = \arg\min_{\hat{\boldsymbol{x}}} \mathbb{E}_{x|y}\left[(\boldsymbol{x} - \hat{\boldsymbol{x}})^T (\boldsymbol{x} - \hat{\boldsymbol{x}}) | \boldsymbol{y} \right] = \mathbb{E}[\boldsymbol{x}|\boldsymbol{y}]$$

Finds expected value/mean of $\boldsymbol{x}$ conditioned on $\boldsymbol{y}$. Typically more robust to outliers and modelling errors as the entire distribution is considered, but bad for bimodal distributions.

Undesirable cases shown below: Left: ML picks x_1, which is not robust. Right: MMSE picks x_1, which has zero probability.



13 Properties of Gaussian Distributions

1-dimensional: $f(x|\mu,\sigma^2) = \dfrac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\dfrac{1}{2}\dfrac{(x-\mu)^2}{\sigma^2} \right)$

N-dimensional: $f(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \dfrac{1}{\sqrt{(2\pi)^N \det \boldsymbol{\Sigma}}} \exp\left(-\dfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}-\boldsymbol{\mu}) \right)$

Central Limit Theorem: Let $y_1, \dots, y_n$ be a sequence of n independent random variables drawn from (possibly different) distributions with finite population mean μ and variance σ^2 and let $\bar{y} = y_1 + \cdots + y_n$, then for $n \to \infty$, $\bar{y} \sim \mathcal{N}(n\mu, n\sigma^2)$.

A multivariate Gaussian $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be partitioned:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix}, \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix} \implies \boldsymbol{x} \sim f(\boldsymbol{x}_1, \boldsymbol{x}_2)$$

Then $f(\boldsymbol{x}_1) = \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}), f(\boldsymbol{x}_2) = \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22})$, and

$$f(\boldsymbol{x}_1|\boldsymbol{x}_2) = \mathcal{N}(\boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2), \boldsymbol{\sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21})$$

Gaussians are preserved in any linear transformation:

$$\boldsymbol{y}_1 \sim \mathcal{N}(\boldsymbol{a}_1, \boldsymbol{B}_1), \boldsymbol{y}_2 \sim \mathcal{N}(\boldsymbol{a}_2, \boldsymbol{B}_2)$$
$$\implies \boldsymbol{C}_1\boldsymbol{y}_1 + \boldsymbol{C}_2\boldsymbol{y}_2 \sim \mathcal{N}(\boldsymbol{C}_1\boldsymbol{a}_1 + \boldsymbol{C}_2\boldsymbol{a}_2, \boldsymbol{C}_1^T\boldsymbol{B}_1\boldsymbol{C}_1 + \boldsymbol{C}_2^T\boldsymbol{B}_2\boldsymbol{C}_2)$$

For nonlinear $\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x})$, if $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$, then $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{g}(\boldsymbol{\mu}_x), \boldsymbol{A}\boldsymbol{\Sigma}_x\boldsymbol{A}^T)$ (approximately) where $\boldsymbol{A}$ is the Jacobian of $\boldsymbol{g}(\boldsymbol{x})$ at $\boldsymbol{\mu}_x$, if $\boldsymbol{g}$ is roughly linear within $\pm 3\sigma$ of $\boldsymbol{\mu}_x$. $y \sim \mathcal{N}(g(\mu_x), a^2\sigma_x^2)$ in 1D where $a = g'(\mu_x)$.

The product of multiple Gaussians is a Gaussian if normalized, with

$$\mathcal{N}(\mu, \sigma^2) = \beta \prod_i \mathcal{N}(\mu_i, \sigma_i^2) \qquad \frac{1}{\sigma^2} = \sum_i \frac{1}{\sigma_i^2} \qquad \frac{\mu}{\sigma^2} = \sum_i \frac{\mu_i}{\sigma_i^2}$$

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \beta \prod_i \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \qquad \boldsymbol{\Sigma}^{-1} = \sum_i \boldsymbol{\Sigma}_i^{-1} \qquad \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} = \sum_i \boldsymbol{\Sigma}_i^{-1}\boldsymbol{\mu}_i$$