1 Moving Average Kernel

How do we instantaneously apply a moving average with window size m to a signal $x[k]$? Start with a discrete time series

$$x[k] \quad \text{for} \quad k = 0, ..., K-1 \tag{1}$$

By definition, we would calculate the moving average at time step k, $y[k]$, with window size m as

$$y[k] = \frac{1}{m} \sum_{n=0}^{m-1} x[k-n] \tag{2}$$

which is equivalent to convolving with the kernel

$$g[k] = \begin{cases} \frac{1}{m}, & k = 0, ..., K-1, \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

That is,

$$y[k] = (x * g)[k] = \sum_{n=-\infty}^{\infty} x[k-n]g[n] = \frac{1}{m} \sum_{n=0}^{m-1} x[k-n] \tag{4}$$

Note: the moving average preserves the mean because the kernel has finite support and unit area:

$$\sum_{k} g[k] = 1 \tag{5}$$

2 Stochastic Weather Scenario Generation

Let the historical hourly time series be

$$P_{\text{hist}}[k] \geq 0 \tag{6}$$
$$-20 \leq T_{\text{hist}}[k] \leq 50 \tag{7}$$
$$R_{\text{hist}}[k] \geq 0 \tag{8}$$

for $k = 0, ..., K-1$ where K is the length of the growing season in hours, respectively representing precipitation (inches), temperature (°C), and solar radiation (W/m^2). Then, let a stochastic weather scenario be defined by the parameters

$$\theta = (s_P, s_T, s_R, \eta, \mathcal{D}, \mathcal{H}, \mathcal{C}) \tag{9}$$

where

- s_P is the precipitation scaling factor
- s_T is the temperature offset

- s_R is the radiation scaling factor
- η is the relative noise level (a fraction of the historical standard deviation)
- $\mathcal{D}$ is a drought event
- $\mathcal{H}$ is a heat wave event
- $\mathcal{C}$ is a cold snap event

Each of the event types $(\mathcal{D}, \mathcal{H}, \mathcal{C})$ are defined by three parameters:

$$[k_0, \kappa, \iota] \tag{10}$$

where

- k_0 is the hour when the event begins
- κ is the duration of the event in number of hours
- $\iota \in [0, 1]$ is the intensity, with 1 being the most intense

2.1 Global Scaling and Offset

Initialize the synthetic environmental disturbance time series after applying the global adjustments (s_P, s_T, s_R) for $k = 0, ..., K-1$ as below

$$\begin{aligned} P^{(1)}[k] &= s_P P_{\text{hist}}[k] && \text{(scaling)} \\ T^{(1)}[k] &= T_{\text{hist}}[k] + s_T && \text{(offset)} \\ R^{(1)}[k] &= s_R R_{\text{hist}}[k] && \text{(scaling)} \end{aligned} \tag{11}$$

2.2 Add White Noise

If $\eta > 0$, draw independent white noise sequences

$$\varepsilon_T[k] \sim \mathcal{N}(0, (\eta\sigma_T)^2) \tag{12}$$

$$\varepsilon_R[k] \sim \mathcal{N}(0, (\eta\sigma_R)^2) \tag{13}$$

where σ_T and σ_R are the standard deviations of the historical temperature and radiation time series, respectively.

We then smooth the white noise with a moving average over $m_T = 24$ hours for temperature and $m_R = 12$ hours for radiation, as we will only apply the noise to the radiation during the day time (when it is not as close to zero).

$$\bar{\varepsilon}_T = \frac{1}{m_T} \sum_{n=0}^{m_T - 1} \varepsilon_T[k-n] \tag{14}$$

$$\bar{\varepsilon}_R = \frac{1}{m_R} \sum_{n=0}^{m_R - 1} \varepsilon_R[k-n] \tag{15}$$

We then add that noise to the signals:

$$T^{(2)}[k] = T^{(1)}[k] + \bar{\varepsilon}_T[k] \tag{16}$$

$$R^{(2)}[k] = R^{(1)}[k] + \mathcal{M}_{\text{day}}\bar{\varepsilon}_R[k] \tag{17}$$

where

$$\mathcal{M}_{\text{day}} = \{k | R_{\text{hist}}[k] > R_{\text{day}}\} \tag{18}$$

and we choose the threshold $R_{\text{day}} = 10$ W/m^2.

Precipitation is unchanged in this step, so

$$P^{(2)}[k] = P^{(1)}[k] \tag{19}$$

If $\eta = 0$, then we let $(P^{(2)}, T^{(2)}, R^{(2)}) = (P^{(1)}, T^{(1)}, R^{(1)})$.

2.3 Drought Injection

A drought event is

$$\mathcal{D} = [k_0, \kappa, \iota] \tag{20}$$

For each event, we define the affected hourly index set as

$$\mathcal{I} = \{k_0, k_0 + 1, ..., \min(k_0 + \kappa - 1, K - 1)\} \tag{21}$$

and then apply the intensity scaling to those indices with

$$P^{(3)}[k] \leftarrow (1 - \iota)P^{(2)}[k] \quad \text{for all } k \in \mathcal{I} \tag{22}$$

Temperature and radiation are unchanged in this step, so

$$T^{(3)}[k] = T^{(2)}[k] \quad \text{and} \quad R^{(3)}[k] = R^{(2)}[k] \tag{23}$$

2.4 Heat Wave Injection

A heat wave event is

$$\mathcal{H} = [k_0, \kappa, \iota] \tag{24}$$

where k_0 and κ have the same meanings they did for the drought event, but now ι is the peak temperature add (an offset rather than a scaling factor).

We then construct a ramp-up, hold, and ramp-down for the heat wave. Let

$$\kappa_{\text{ramp}} = \max(1, 0.1\kappa) \tag{25}$$

i.e. either one tenth of the heat wave duration or at least one hour. Then, let

$$\kappa_{\text{hold}} = \kappa - 2\kappa_{\text{ramp}} \tag{26}$$

to account for the ramp-up and ramp-down.

We can then define

- ramp-up: $w_{\text{up}}[k] = \dfrac{k}{\kappa_{\text{ramp}} - 1}$ for $k = 0, ..., \kappa_{\text{ramp}} - 1$
- ramp-up: $w_{\text{hold}}[k] = 1$
- ramp-down: $w_{\text{down}}[k] = \dfrac{k}{\kappa_{\text{ramp}} - 1}$ for $k = 0, ..., \kappa_{\text{ramp}} - 1$

and concatenate to form the heat wave window

$$w_{\text{heat}} = [w_{\text{up}}, w_{\text{hold}}, w_{\text{down}}] \tag{27}$$

Applying the heat wave to the temperature time series, we obtain

$$T^{(4)}[k] \leftarrow T^{(3)}[k] + \iota w_{\text{heat}}[k - k_0] \tag{28}$$

for $k = k_0, k_0 + 1, ..., \min(k_0 + \kappa - 1, K - 1)$.
Precipitation and radiation are unchanged in this step, so

$$P^{(4)}[k] = P^{(3)}[k] \quad \text{and} \quad R^{(4)}[k] = R^{(3)}[k] \tag{29}$$

2.5 Cold Snap Injection

A cold snap event is

$$\mathcal{C} = [k_0, \kappa, \iota] \tag{30}$$

where k_0 and κ have the same meanings they did for the heat wave event, but ι is now the magnitude of the lowest temperature drop. A cold snap window is constructed in the same manner as the heat wave window and the temperature time series becomes

$$T^{(4)}[k] \leftarrow T^{(4)}[k] - \iota w_{\text{heat}}[k - k_0] \tag{31}$$

for $k = k_0, k_0 + 1, ..., \min(k_0 + \kappa - 1, K - 1)$. Note: we have used $T^{(4)}[k]$ on the righthand side because we want to reflect that a heat wave may have already been applied.

2.6 Clipping to Physical Bounds

Finally, we check to ensure that the transformations above have not violated the bounds on the input disturbances specified in equations 8. If they have, we clip the values as below

$$P_{\text{syn}}[k] = \max(0, P^{(4)}[k]) \tag{32}$$

$$T_{\text{syn}}[k] = \min(50, \max(-20, T^{(4)}[k])) \tag{33}$$

$$R_{\text{syn}}[k] = \max(0, R^{(4)}[k]) \tag{34}$$

for all $k \in \{0, ..., K - 1\}$.

3 Extremity Index

Extremity from precipitation:

$$\mathcal{E}_{\text{precip}} = 2|1 - s_P| \tag{35}$$

So, $s_P = 1$ is the threshold for a drought season vs. a wet season with $s_P < 1$ indicating drought and $s_P > 1$ indicating wet.

Extremity from temperature:

$$\mathcal{E}_{\text{temp}} = \frac{|s_T|}{5} \tag{36}$$

indicating that ±5°C is the threshold for extreme/not extreme temperature.

Extremity from a drought event:

$$\mathcal{E}_{\text{drought}} = \sum_{(k_0,\kappa,\iota)\in\mathcal{D}} \frac{\kappa}{500}\iota \tag{37}$$

denoting that 500 hours (21 days) constitutes the threshold for long vs. short drought.

Extremity from a heat wave or cold snap event:

$$\mathcal{E}_{\text{heat}} = \sum_{(k_0,\kappa,\iota)\in\mathcal{H}} \frac{\kappa}{200}\frac{\iota}{5} \tag{38}$$

$$\mathcal{E}_{\text{cold}} = \sum_{(k_0,\kappa,\iota)\in\mathcal{C}} \frac{\kappa}{200}\frac{\iota}{5} \tag{39}$$

indicating that a long heat wave or cold snap is considered to be > 500 hours and temperatures are considered to be extreme if 5°C higher or lower than typical.

The aggregate extremity score is then

$$\mathcal{E} = \mathcal{E}_{\text{precip}} + \mathcal{E}_{\text{temp}} + \mathcal{E}_{\text{drought}} + \mathcal{E}_{\text{heat}} + \mathcal{E}_{\text{cold}} \tag{40}$$

4 Constrained Finite Time Optimal Control

Discrete-time optimal control is concerned with choosing an optimal input sequence over the horizon K

$$\mathcal{U}_{0\to K} = \{u[k]\} \quad \text{for} \quad k = 0, ..., K-1 \tag{41}$$

with respect to some objective function over a finite or infinite time horizon in order to apply it to a system with a given initial state $x[0]$.

The objective function is often defined as a sum of stage costs $q(x[k], u[k])$ and when the horizon has finite length, a terminal cost $p(x[K])$. That is

$$J_{0\to K}(x[0], \mathcal{U}_{0\to K}) = p(x[K]) + \sum_{n=0}^{K-1} q(x[n], u[n]) \tag{42}$$

where the states $\mathcal{X}_{0\to K} = \{x[k]\} \quad \text{for} \quad k = 0, ..., K-1$ must satisfy the initial condition and system dynamics

$$\begin{cases} x[0] = x_0 \\ x[k+1] = g(x[k], u[k]) \quad \text{for } k = 0, ..., K-1 \end{cases} \tag{43}$$

and there may be other state or input constraints formulated as inequalities

$$h(x[k], u[k]) \leq 0 \quad \text{for} \quad k = 0, ..., K-1 \tag{44}$$

In the finite horizon case, there may also be a terminal constrain requiring the final state to lie in some terminal set $x[K] \in \mathcal{X}_{\text{final}}$.

In our specific case, we define a daily horizon K_d in hours and construct the problem as a minimization

$$\begin{gathered} J^*_{0\to K_d}(x[0]) = \min_{\mathcal{U}_{0\to K_d}} \; J_{0\to K_d}(x[0], \mathcal{U}_{0\to K_d}) \quad \text{s.t.} \\ \begin{cases} x[0] = x_0 \\ x[k+1] = g(x[k], u[k]) \quad \text{for } k = 0, ..., K-1 \\ x \in \mathbb{R}^+ \\ u \in \mathcal{U} \end{cases} \end{gathered} \tag{45}$$

where the stage cost is

$$q(x[k], u[k]) = w_w \left(\frac{u_w[k]}{w_{\text{typ}}} \right)^2 + w_f \left(\frac{u_f[k]}{f_{\text{typ}}} \right)^2 + w_{\delta w}(\delta_w[k])^2 + w_{\delta f}(\delta_f[k])^2 \tag{46}$$

and the terminal cost is

$$p(x[K_d]) = w_h \frac{x_h[K_d]}{k_h} + w_A \frac{x_A[K_d]}{k_A} + w_P \frac{x_P[K_d]}{k_P} \tag{47}$$

There is no terminal set constraint because we want the plant to grow as much as possible. The terms in the stage cost have been squared in order to encourage sparsity in actuation, more similar to what a farmer might actually implement.

5 Automatic Relevance Determination for Kernel Fitting

Given an objective function $f : \mathcal{X} \subset \mathbb{R}^d \to \mathbb{R}$, if we evaluate at a set of n points $\mathcal{X}_n = \{x_i\}_{i=1}^n$, then we can construct the vector

$$\mathbf{f} = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix} \tag{48}$$

Given this collection of points, we can construct a Gaussian Process, which is a finite collection of points that has a joint Gaussian distribution which encodes properties like smoothness, length scales, and periodicity. Mathematically, we describe a Gaussian Process with

$$\mathbf{f} \sim \mathcal{GP}(\mathbf{m}(\mathbf{x}), g(\mathbf{x}, \mathbf{x}')) \tag{49}$$

where $\mathbf{m}(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]\mathbf{1}_n$ is the expected value of the random function $f(\mathbf{x})$ and $g(\mathbf{x}, \mathbf{x}')$ is a kernel which describes the structure of the distribution. It is typical to use the radial basis function kernel

$$g(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left\{ -\frac{1}{2} \sum_{j=1}^{d} \frac{(x_j - x_j')^2}{l_j^2} \right\} \tag{50}$$

where

- σ_f^2 is the variance of the function $f(x)$
- $\{l_j\}_{j=0}^d$ are the length scales for the d dimensions of $f(\mathbf{x})$

The function is sensitive in dimension j is l_j is small and varies slowly in dimension j is l_j is large. If $l_j \to \infty$ then dimension j is irrelevant. That these properties are encoded in the $\mathcal{GP}$ kernel is what defines Automatic Relevance Determination (ARD).

Given the training inputs $\mathcal{X}_n$, we can build the Gram matrix (kernel matrix), which is a set of inner product vectors

$$\mathbf{K}_f \in \mathbb{R}^{n \times n}, \quad (\mathbf{K}_f)_{ij} = g(x_i, x_j) \tag{51}$$

and represents what the $\mathcal{GP}$ prior thinks the covariance between $f(x_i)$ and $f(x_j)$ should be.

If observations are noisy, then

$$y_i = f(x_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_n^2) \tag{52}$$

Thus, the mean of y is

$$\mathbb{E}[\mathbf{y}] = \mathbb{E}[\mathbf{f}] + \mathbb{E}[\boldsymbol{\epsilon}] = \mathbf{m} \tag{53}$$

and the covariance of y is

$$\mathrm{Cov}(\mathbf{y}) = \mathrm{Cov}(\mathbf{f}) + \mathrm{Cov}(\boldsymbol{\epsilon}) = \mathbf{K}_f + \sigma_n^2\mathbf{I} \triangleq \mathbf{K} \tag{54}$$

So, $\mathbf{y} \sim \mathcal{N}(\mathbf{m}, \mathbf{K})$. That implies that the marginal likelihood is the multivariate normal density

$$p(\mathbf{y}|\mathcal{X}_n, \boldsymbol{\theta}) = \frac{1}{(2\pi)^{n/2}|\mathbf{K}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{y}-\mathbf{m})^T\mathbf{K}^{-1}(\mathbf{y}-\mathbf{m})\right\} \tag{55}$$

where the parameters $\boldsymbol{\theta}$ are the kernel variance, the dimensions of $\mathbf{x}$, and the variance of the samples:

$$\boldsymbol{\theta} = \begin{bmatrix} \sigma_f^2 \\ l_1 \\ \vdots \\ l_d \\ \sigma_n^2 \end{bmatrix} \tag{56}$$

Taking the log of the likelihood

$$\log p(\mathbf{y}|\mathcal{X}_n, \boldsymbol{\theta}) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}(\mathbf{y}-\mathbf{m})^T\mathbf{K}^{-1}(\mathbf{y}-\mathbf{m}) - \frac{1}{2}\log|\mathbf{K}| \tag{57}$$

Letting $\boldsymbol{\delta} = \mathbf{y} - \mathbf{m}$ and $\boldsymbol{\alpha} = \mathbf{K}^{-1}\boldsymbol{\delta}$, we can then write the log likelihood as

$$\mathcal{L}(\boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{\delta}^T\mathbf{K}^{-1}\boldsymbol{\delta} - \frac{1}{2}\log|\mathbf{K}| - \frac{n}{2}\log(2\pi) \tag{58}$$

Knowing that

$$\frac{\partial \mathbf{K}^{-1}}{\partial \theta_j} = -\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_j}\mathbf{K}^{-1} \tag{59}$$

and

$$\frac{\partial}{\partial \theta_j}\log|\mathbf{K}| = \mathrm{tr}\left(\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta + j}\right) \tag{60}$$

and the identity $\mathbf{a}^T\mathbf{B}\mathbf{a} = \mathrm{tr}(\mathbf{B}\mathbf{a}\mathbf{a}^T)$, as well as that $\mathbf{K}^T = \mathbf{K}$, then the gradient of the log likelihood is

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \theta_j} &= -\frac{1}{2}\frac{\partial}{\partial \theta_j}\left(\boldsymbol{\delta}^T\mathbf{K}^{-1}\boldsymbol{\delta}\right) - \frac{\partial}{\partial \theta_j}\left(\frac{1}{2}\log|\mathbf{K}|\right) - \cancelto{0}{\frac{\partial}{\partial \theta_j}\left(\frac{n}{2}\log(2\pi)\right)} && (61)\\
&= -\frac{1}{2}\boldsymbol{\delta}^T\frac{\partial \mathbf{K}^{-1}}{\partial \theta_j}\boldsymbol{\delta} - \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_j}\right) && (62)\\
&= -\frac{1}{2}\boldsymbol{\delta}^T\left(-\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_j}\mathbf{K}^{-1}\right)\boldsymbol{\delta} - \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_j}\right) && (63)\\
&= \frac{1}{2}\boldsymbol{\alpha}^T\frac{\partial \mathbf{K}}{\partial \theta_j}\boldsymbol{\alpha} - \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_j}\right) && (64)
\end{aligned}$$

$$= \frac{1}{2}\mathrm{tr}\left(\frac{\partial \mathbf{K}}{\partial \theta_j}\boldsymbol{\alpha}\boldsymbol{\alpha}^T\right) - \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\frac{\partial \mathbf{K}}{\partial \theta_j}\right) \tag{65}$$

$$= \frac{1}{2}\mathrm{tr}\left[(\boldsymbol{\alpha}\boldsymbol{\alpha}^T - \mathbf{K}^{-1})\frac{\partial \mathbf{K}}{\partial \theta_j}\right] \tag{66}$$

For a Gaussian kernel, as in equation 50, say

$$(\mathbf{K}_f)_{ab} = \sigma_f^2 \exp\left\{-\frac{1}{2}\sum_{j=1}^{d}\frac{\Delta_{ab}^{(j)}}{l_j^2}\right\} \tag{67}$$

then

$$\frac{\partial(\mathbf{K}_f)_{ab}}{\partial \sigma_f^2} = \exp\left\{-\frac{1}{2}\sum_{j=1}^{d}\frac{\Delta_{ab}^{(j)}}{l_j^2}\right\} = \frac{1}{\sigma_f^2}(\mathbf{K}_f)_{ab} \tag{68}$$

and, if we define

$$E_{ab} = \frac{1}{2}\sum_{j=1}^{d}\frac{\Delta_{ab}^{(j)}}{l_j^2} \tag{69}$$

then

$$\frac{\partial E_{ab}}{\partial l_j} = \frac{\Delta_{ab}^{(j)}}{l_j^3} \tag{70}$$

By the chain rule,

$$\frac{\partial(\mathbf{K}_f)_{ab}}{\partial l_j} = \frac{\partial(\mathbf{K}_f)_{ab}}{\partial E_{ab}} \cdot \frac{\partial E_{ab}}{\partial l_j} = (\mathbf{K}_f)_{ab} \cdot \frac{\Delta_{ab}^{(j)}}{l_j^3} \tag{71}$$

Finally, since $\mathbf{K} = \mathbf{K}_f + \sigma_n^2\mathbf{I}$,

$$\frac{\partial(\mathbf{K}_f)_{ab}}{\partial \sigma_n^2} = \mathbf{I} \tag{72}$$

With the partial derivatives

$$\frac{\partial(\mathbf{K}_f)_{ab}}{\partial \sigma_f^2} = \frac{1}{\sigma_f^2}(\mathbf{K}_f)_{ab}, \quad \frac{\partial(\mathbf{K}_f)_{ab}}{\partial l_j} = (\mathbf{K}_f)_{ab} \cdot \frac{\Delta_{ab}^{(j)}}{l_j^3} \quad \frac{\partial(\mathbf{K}_f)_{ab}}{\partial \sigma_n^2} = \mathbf{I} \tag{73}$$

we can construct the full gradient of the log likelihood $\nabla_{\boldsymbol{\theta}}\mathcal{L}$ and, using a method like L-BFGS or Adam[1], determine the parameters $\boldsymbol{\theta}$ that maximize the log likelihood i.e.

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \log p(\mathbf{y}|\mathcal{X}_n, \boldsymbol{\theta}) \tag{74}$$

[1] L-BFGS = Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm, a second-order quasi-Newton method; Adam = Adaptive Moment Estimation, a first-order adaptive method that is an extension to Stochastic Gradient Descent.

Conditioning the joint Gaussian with the optimal $\boldsymbol{\theta}$ yields the posterior $\mathcal{GP}$ with mean

$$\boldsymbol{\mu}(\mathbf{x}) = \mathbf{m}(\mathbf{x}) + g(\mathbf{x}, \mathcal{X}_n)(\mathbf{K}^*)^{-1}(\mathbf{y} - \mathbf{m}(\mathcal{X}_n)) \tag{75}$$

and variance

$$\boldsymbol{\sigma}^2(\mathbf{x}) = g(\mathbf{x}, \mathbf{x}) - (g(\mathbf{x}, \mathcal{X}_n))^T(\mathbf{K}^*)^{-1}g(\mathcal{X}_n, \mathbf{x}) \tag{76}$$

At this point, we have a belief distribution over the objective surface, where $\mu(x)$ (the mean evaluated at a single point) is our best guess for $f(\mathbf{x})$ and the variance is the uncertainty due to lack of data.

6 Gaussian Process Bayesian Optimization

Bayesian Optimization is in the "high cost of evaluation, low number of evaluations" regime of optimization. We are not fitting a model and then optimizing it – we are maintaining uncertainty and deciding what to do next, and the posterior variance is the decision signal.

In Bayesian Optimization (BO), we start with an objective function

$$f : \mathcal{X} \subset \mathbb{R}^d \to \mathbb{R} \tag{77}$$

that we want to maximize (or minimize).

The catch:

- Evaluating $f(x)$ is expensive
- We do not have gradients
- We can only afford tens or hundreds of evaluations
- $f(x)$ might be noisy

Typical applications:

- Hyperparameter tuning
- Physical experiments
- Black box simulations

Core idea: Treat the unknown objective function as a random function, update beliefs as data is collected, choose the next evaluation point by trading off between exploration and exploration. That is, we do not assume a parametric form for $f(x)$ – we simply assume structure.

In BO, we place a prior over functions, assuming that $f(x)$ is a Gaussian Process, as in equation 49. We then observe n noisy evaluations

$$y_i = f(x_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_n^2) \tag{78}$$

and then condition the $\mathcal{GP}$ prior on the observed data $\mathcal{D}_n = \{(x_i, y_i)\}_{i=0}^n$ to yield the posterior $\mathcal{GP}$:

$$f(x)|\mathcal{D}_n \sim \mathcal{N}(\mu_n(x), \sigma_n^2(x)) \tag{79}$$

At this point, everything we know about $f(x)$ is contained in the two scalars $\mu(x)$ and $\sigma_n(x)$, which we find with Automatic Relevance Determination.

We must now make a decision about where to sample next, and we do that by maximizing an "acquisition function." A typical choice for the acquisition function is the "expected improvement," or the expected value of the improvement, defined as

$$I(x) = \max(0, f(x) - f^+) \tag{80}$$

where

$$f^+ = \max_{i \leq n} y_i \tag{81}$$

Key idea: Improvement $I(x)$ is random because $f(x)$ is random. Our goal is to choose x to maximize the expected improvement. Writing the expected improvement as an integral using the definition of expected value

$$\mathbb{E}[I(x)] = \int_{-infty}^{infty} I(x) p(I(x))\, dI \tag{82}$$

and noting that $p(I(x)) = p(f(x))$ and the improvement is zero when $f \leq f^+$, the expected improvement becomes

$$\mathbb{E}[I(x)] = \int_{f^+}^{\infty} (f - f^+) p(f|x, \mathcal{D}_n)\, df \tag{83}$$

Since we assume $f(x)$ is a Gaussian Process,

$$p(f) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ \frac{(f-\mu)^2}{\sigma^2} \right\} \tag{84}$$

By substitution,

$$\mathbb{E}[I(x)] = \int_{f^+}^{\infty} (f - f^+) \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2} \frac{(f-\mu)^2}{\sigma^2} \right\} df \tag{85}$$

Defining

$$z = \frac{f - \mu}{\sigma} \tag{86}$$

the lower integration bound and differential become

$$z_0 = \frac{f^+ - \mu}{\sigma} \quad \text{and} \quad dz = \frac{1}{\sigma} df \tag{87}$$

and the expected improvement becomes

$$\mathbb{E}[I(x)] = \int_{z_0}^{\infty} (z\sigma + \mu - f^+) \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2} z^2 \right\} \sigma\, dz \tag{88}$$

Letting the standard normal be

$$\phi(z) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{1}{2}z^2\right\} \tag{89}$$

the expected improvement becomes

$$\mathbb{E}[I(x)] = \int_{z_0}^{\infty} (z\sigma + \mu - f^+)\phi(z)\, dz \tag{90}$$

Since

$$\int_{z_0}^{\infty} \phi(z)\, dz = \frac{1}{2}\text{erf}\left(\frac{z_0}{\sqrt{2}}\right) \triangleq \Phi(z_0) \tag{91}$$

and

$$\int_{z_0}^{\infty} z\phi(z)\, dz = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{1}{2}z_0^2\right\} = \phi(z_0) \tag{92}$$

then by substitution

$$\begin{aligned}
\mathbb{E}[I(x)] &= \int_{z_0}^{\infty} (z\sigma + \mu - f^+)\phi(z)\, dz && (93)\\
&= \sigma \int_{z_0}^{\infty} z\phi(z)\, dz + (\mu - f^+) \int_{z_0}^{\infty} \phi(z)\, dz && (94)\\
&= \underbrace{\sigma(x)\phi(z_0)}_{\text{exploration}} + \underbrace{(\mu(x) - f^+)\Phi(z_0)}_{\text{exploitation}} && (95)
\end{aligned}$$

From which we see that exploration is not added heuristically – it falls out of probability theory.

Key observations:

- If $\sigma = 0$, $\mathbb{E}[I(x)] = \max(0, \mu - f^+)$, which is in line with expectation
- If $\mu << f^+$, $\mathbb{E}[I(x)] \neq 0$ if σ is large, so even though the current guess is off, there is still chance that "good" solutions can be found

Now, a standard gradient descent method can be used to solve for

$$x^* = \arg\max_{x\in\mathcal{X}} \mathbb{E}[I(x)] \tag{96}$$

With the next choice of data point to sample ($x_{n+1} = x^*$), we can perform the "expensive" evaluation

$$y_{n+1} = f(x_{n+1}) \tag{97}$$

and update the dataset

$$\mathcal{D}_n \leftarrow \mathcal{D}_{n+1} = \mathcal{D}_n \cup (x_{n+1}, y_{n+1}) \tag{98}$$

and return

$$\max_i y_i \tag{99}$$

We then perform this optimization loop either until convergence or some set number of data points has been collected.

7 Bayesian Optimization with Tree-structured Parzen Estimation

In Gaussian Process Bayesian Optimization, the objective is assumed to be a Gaussian Process

$$f(x) \sim \mathcal{GP}(m(x), g(x, x')) \tag{100}$$

where x are the hyperparameter candidates and $f(x)$ is the associated performance metric (like revenue in our application), and you get a posterior over functions

$$p(f(x)|\mathcal{D}_n) \sim \mathcal{N}(\boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\sigma}_n^2(\mathbf{x})) \tag{101}$$

that yields a mean surface, an uncertainty surface, and an explicit correlation between sampled points. One then defines an acquisition function like the expected improvement and can maximize to obtain the next data point to sample

$$x_{n+1} = \arg\max_x \mathbb{E}(\mu(x), \sigma(x)) \tag{102}$$

which is an explicit tradeoff between exploration and exploitation based in probability theory.

In Bayesian Optimization with Tree-structured Parzen Estimation (TPE), instead of directly modeling the objective function's output probability $p(y|x)$ (the posterior), TPE models do the reverse: they model $p(x|y)$ with two density functions.

TPE splits the observed hyperparameter configurations into two groups based on a chosen quantile γ of their performance scores (e.g. "good" values = top 25%). Then, say

- $l(x)$: density function for hyperparameter values that resulted in a "good" objective score ("less than threshold").
- $g(x)$: density function for hyperparameter values that resulted in a "bad" objective score ("greater than threshold").

Kernel Density Estimation (also known as non-parametric Parzen windows) is used to approximate these density functions.

In TPE, the expected improvement is defined as $l(x)/g(x)$ so that the next hyperparameter sample set has a good change of being "good."

To model $p(x|y)$ instead of $p(y|x)$, we need Bayes' rule:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} \tag{103}$$

We then define a threshold y^- such that

$$P(y \leq y^-) = \gamma, \quad \gamma \in (0, 1) \tag{104}$$

but more typically, we constrain the quartile options to something like $\gamma \in (0.1, 0.25)$.

We can then define the two conditional densities as

$$l(x) = p(x|y \leq y^-) \tag{105}$$

$$g(x) = p(x|y > y^-) \tag{106}$$

The expected improvement is then

$$\mathbb{E}[I(x)] = \int_{-\infty}^{y^-} (y^- - y)p(y|x)\,dy \quad \text{(minimization)} \tag{107}$$

Substituting for Bayes' rule,

$$\mathbb{E}[I(x)] = \int_{-\infty}^{y^-} (y^- - y)\frac{p(x|y)p(y)}{p(x)}\,dy \tag{108}$$

TPE assumes that for $y \leq y^-$, $p(x|y) \approx l(x)$, so then

$$\mathbb{E}[I(x)] = \frac{l(x)}{p(x)} \int_{-\infty}^{y^-} (y^- - y)p(y)\,dy \tag{109}$$

Since the integrand is independent of x, it is a constant, and thus we know

$$\mathbb{E}[(I(x))] \propto \frac{l(x)}{p(x)} \tag{110}$$

By the law of total probability, we know

$$p(x) = \gamma l(x) + (1 - \gamma)g(x) \tag{111}$$

so therefore

$$\mathbb{E}[(I(x))] \propto \frac{l(x)}{\gamma l(x) + (1 - \gamma)g(x)} \tag{112}$$

Since γ is fixed, maximizing $\mathbb{E}[I(x)]$ is equivalent to maximizing $l(x)/g(x)$. Thus, we deem $l(x)/g(x)$ to be the TPE acquisition function.

TPE is computationally faster than Gaussian Process BO and does well with mixed hyperparameter spaces (continuous, discrete, categorical). As MPC solvers often produce discontinuities, failures, and flat regions, and we have constrained daily horizon to be an integer while all other hyperparameters are continuous values, TPE is a better choice for our application.

8 Kernel Density Estimation

Idea: Place a kernel around each data point, sum the kernels, normalize by the number of points, and smooth with some window size h.

Given samples $\{x_i\}_{i=1}^n$ the kernel estimator is

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n} g(x, x') \tag{113}$$

Usually, the kernel choice is Gaussian:

$$g(x, x') = \mathcal{N}(x|x', h^2) \tag{114}$$

So for one dimension,

$$\hat{p}(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi h^2}} \exp\left\{-\frac{1}{2}\frac{(x-x')^2}{h^2}\right\} \tag{115}$$

TPE uses the naive Bayes' assumption that dimensions are independent within both the good and bad sets:

$$l(x) \approx \Pi_{j=1}^{d} l_j(x_j) \quad \text{and} \quad g(x) \approx \Pi_{j=1}^{d} g_j(x_j) \tag{116}$$

Now, TPE does not optimize $l(x)/g(x)$ directly. Instead, it samples many candidates from $l(x)$, then evaluates $l(x)/g(x)$ for each, and picks the best x.

The tree-structured part of the Parzen estimator comes from the fact that in real hyperparameter spaces, some parameters only exist if other take on certain values i.e. the parameters are jointly distributed:

$$p(x) = p(x_1)p(x_2|x_2)p(x_3|x_2)... \tag{117}$$

in what we call a tree of conditional distributions.