Recap: Deterministic MPC

- MPC solves a N-step lookahead problem (called **Optimal Control** problem) in a receding horizon fashion, in order to approximate the infinite-horizon DP:

Terminal Cost Approximation

$$
\begin{aligned}
J_0(\bar{x}_0) = \min_{X,U} \quad & \hat{J}_N(x_N) + \sum_{i=0}^{N-1} g_i(x_i, u_i) \\
\text{s.t.} \quad & x_{i+1} = f_i(x_i, u_i), \quad \forall i \in \{0, 1, 2, \ldots\} \\
& x_i \in \mathcal{X}, \quad \forall i \in \{0, 1, 2, \ldots\} \\
& u_i \in \mathcal{U}, \quad \forall i \in \{0, 1, 2, \ldots\} \\
& x_0 = \bar{x}_0 \\
& x_N \in \mathcal{X}_f
\end{aligned}
$$

Terminal set constraint

Introducing Disturbances

- Now let's re-introduce disturbances to our linear system:

$$x_{k+1} = Ax_k + Bu_k + w_k$$

- Where the disturbance vector w_k will belong to some polytope $\mathcal{W}$:

$$w_k \in \mathcal{W} = \left\{ w \in \mathbb{R}^n : H_w w \leq h_w \right\}, \forall k \geq 0$$

- With the addition of uncertainty, it is not straightforward on how to change the Optimal Control Problem that the MPC algorithm needs to solve at every step.

Recap: Robust Set Computation

- Recall our definition of **robust** predecessor set:

$$P(\mathcal{X}, \mathcal{W}) = \left\{ x \in \mathbb{R}^n : \ \exists u \in \mathcal{U} \text{ s.t.: } Ax + Bu + w \in \mathcal{X}, \forall w \in \mathcal{W} \right\}$$

- And our summary on how to compute them:

	$x_{k+1} = Ax_k + w_k$	$x_{k+1} = Ax_k + Bu_k + w_k$
$P(\mathcal{X})$	$\mathcal{X} \circ A$	$(\mathcal{X} \oplus (-B \circ \mathcal{U})) \circ A$
$P(\mathcal{X}, \mathcal{W})$	$(\mathcal{X} \ominus \mathcal{W}) \circ A$	$((\mathcal{X} \ominus \mathcal{W}) \oplus (-B \circ \mathcal{U})) \circ A$

Recap: Control Invariant Set Computation

- Recall our definition of robust control invariant set for a system $x_{k+1} = Ax_k + Bu_k + w_k$ if:

$$x_0 \in \mathcal{C} \Rightarrow \exists u_k \in \mathcal{U} \text{ s.t.: } x_k \in \mathcal{C}, \, \forall w_k \in \mathcal{W}, \, k \geq 1$$

- In words: “a set is control invariant if once our systems starts from it, there is always a feasible control such that once applied the system inside the set for any possible disturbance value.”

- And the largest of such sets is the maximal control invariant set, and we called it $\mathcal{C}_\infty$.

Recap: Invariant Set Computation

- As we saw, a set $\mathcal{C} \subseteq \mathcal{X}$ is control invariant if and only if:

$$\mathcal{C} \subseteq \mathcal{P}(\mathcal{C}, \mathcal{W})$$

- Then the following algorithm can be used to compute robust control invariant sets:

Algorithm 1 Algorithm for Invariant Set Computation

Input: Linear system matrix A, state constraint set $\mathcal{X}$, and disturbance set $\mathcal{W}$.

1: Let $\Omega_0 = \mathcal{X}$
2: **for** $k = 0, 1, 2, 3...$ **do** (obtaining new samples)
3:     Let: $\Omega_{k+1} = \mathcal{P}(\Omega_k, \mathcal{W}) \cap \Omega_k$
4:     If $\Omega_{k+1} = \Omega_k$, Set $\mathcal{C}_\infty \leftarrow \Omega_{k+1}$; Then break
5: **end for**

Output: The Maximal Robust Control Invariant Set $\mathcal{C}_\infty$

Stochastic Optimal Control

- First we can think of our standard stochastic DP formulation:

$$
\begin{aligned}
J_0(\bar{x}_0) = \min_{X,U} \quad & \mathbb{E}_{w_0,\ldots,w_{N-1}}\left[\hat{J}_N(x_N) + \sum_{i=0}^{N-1} g_i(x_i,u_i)\right] \\
\text{s.t.} \quad & x_{i+1} = f_i(x_i,u_i,w_i), \quad \forall i \in \{0,1,2,\ldots\} \\
& x_i \in \mathcal{X}, \quad \forall i \in \{0,1,2,\ldots\} \\
& u_i \in \mathcal{U}, \quad \forall i \in \{0,1,2,\ldots\} \\
& x_0 = \bar{x}_0 \\
& x_N \in \mathcal{X}_f
\end{aligned}
$$

- This optimization can be tackled by using Stochastic Programming methods:
  - Using scenarios for the disturbances vectors and performing Sample Average Approximation.
  - Using change constraints.
  - We will not focus on solving this.

Robust Optimal Control

- Instead we will solve the following problem:

$$
J_0(\bar{x}_0) = \min_{X,U} \quad \hat{J}_N(x_N) + \sum_{i=0}^{N-1} g_i(x_i, u_i)
$$

$$
\left.\begin{aligned}
\text{s.t. } & x_{i+1} = f_i(x_i, u_i, w_i), \quad \forall i \in \{0, 1, 2, ...\} \\
& x_i \in \mathcal{X}, \quad \forall i \in \{0, 1, 2, ...\} \\
& u_i \in \mathcal{U}, \quad \forall i \in \{0, 1, 2, ...\} \\
& x_0 = \bar{x}_0 \\
& x_N \in \mathcal{X}_f
\end{aligned}\right\} \forall w_i \in \mathcal{W}, \forall i \in \{0, 1, 2, ...\}
$$

- In this problem, the optimization problem itself is “deterministic”, but we want it to be feasible for all possible types of disturbance. That is, we want our solution to be **robust** in face of the disturbances.

Open-Loop vs Closed-Loop

- Recall one of our very first lectures, where we made the distinction between open-loop and closed-loop policies:

Deterministic DP

- Open-loop = Closed-loop
- No need for policies
- Can solve via Forward DP

$$(u_0^*, u_1^*, \ldots, u_{N-1}^*)$$

Stochastic DP

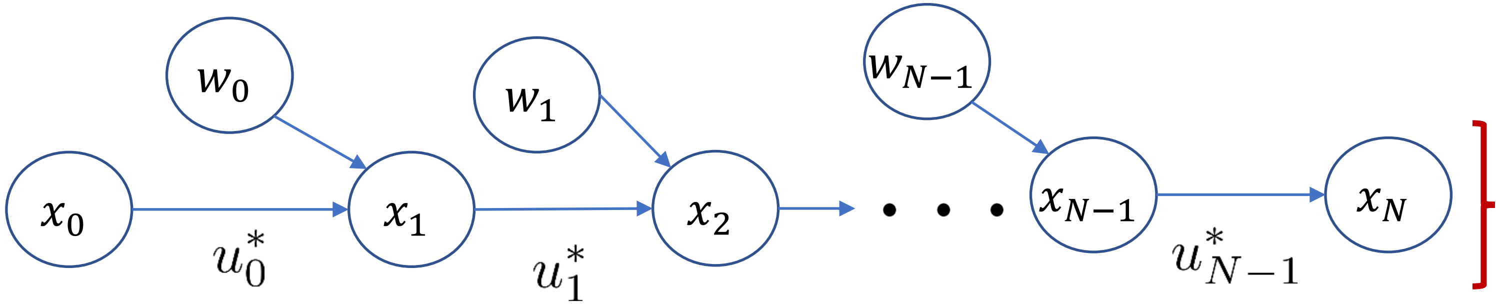
- Open-loop $\neq$ Closed-loop
- We need closed-loop policies
- Must solve via Backward DP

$$\pi^* = (\mu_0^*(x_0), \mu_1^*(x_1), \ldots, \mu_{N-1}^*(x_{N-1}))$$

Open-Loop vs Closed-Loop

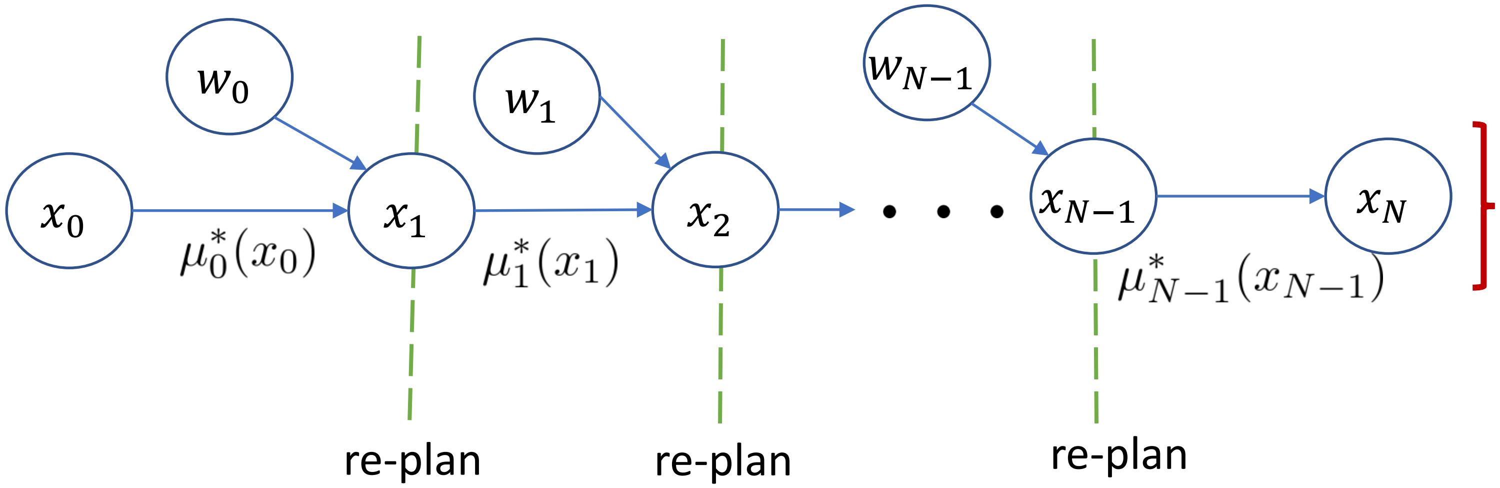
- The key difference here is the **timing** of decisions and when the uncertainty is resolved:

Open-Loop:



You “commit” on decisions at the start

Closed-Loop:



You time your choices after each uncertainty is “revealed”

Robust Optimal Control as a Min-Max game

- One interesting aspect of Robust Optimal Control is that we can view it as Min-Max game.

- Suppose we had no constraints and the problem is reduced to:

$$J_0(\bar{x}_0) = \min_{X,U} \quad \hat{J}_N(x_N) + \sum_{i=0}^{N-1} g_i(x_i, u_i)$$

$$\left.\begin{aligned} \text{s.t.} \quad & x_{i+1} = f_i(x_i, u_i, w_i), \qquad \forall i \in \{0, 1, 2, \ldots\} \\ & x_0 = \bar{x}_0 \end{aligned}\right\} \quad \forall w_i \in \mathcal{W},\ \forall i \in \{0, 1, 2, \ldots\}$$

- Which we can write as:

$$J_0(\bar{x}_0) = \min_{X,U} \quad \max_{W} \quad \hat{J}_N(x_N) + \sum_{i=0}^{N-1} g_i(x_i, u_i)$$

$$\begin{aligned} \text{s.t.} \quad & x_{i+1} = f_i(x_i, u_i, w_i), \qquad \forall i \in \{0, 1, 2, \ldots\} \\ & x_0 = \bar{x}_0 \end{aligned}$$

- Player 1:
  - Selects the control inputs u_k
- Player 2:
  - Selects the worst possible disturbance value w_k

Robust DP as a Min-Max game

- Namely we can write the DP recursion as follows:

$$J_N(x_n) = g_N(x_N)$$

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \Big\{ \max_{w_k \in \mathcal{W}} \big\{ g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \big\} \Big\}, \forall k \in \{0, \ldots, N-1\}$$

- And we note that every time an expectation is required, we would do a maximization instead:
  - Typically this is much more costly, as we approximate the expectation via SAA
  - But the Backward DP algorithm works just fine (as long as the costs are bounded)

Robust DP as a Min-Max game

- If we have constraints on the states, the formulation changes a bit. We can compute a robust set of state constraints, using the idea of precursor sets.

Suppose we have some robust invariant set $\mathcal{O}$ at hand, then we can perform the following backwards recursion:

$$\mathcal{X}_N = \mathcal{O}$$

$$\mathcal{X}_j = \mathcal{P}(\mathcal{X}_{j+1}, \mathcal{W}), \, \forall j \in \{0, ..., N-1\}$$

- The DP recursion would then become:

$$J_N(x_n) = g_N(x_N)$$

$$J_k(x_k) = \min_{u_k \in \bar{U}_k(x_k)} \left\{ \max_{w_k \in \mathcal{W}} \left\{ g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \right\} \right\}, \forall k \in \{0, ..., N-1\}$$

- Where: $\bar{U}_k(x_k) = \{u \in U_k(x_k) : f_k(x_k, u_k, w_k) \in \mathcal{X}_{k+1}, \forall w_k \in \mathcal{W}\}$

Robust Model Predictive Control

- Let's return to our Robust Optimal Control problem

$$
\left.
\begin{aligned}
J_0(\bar{x}_0) = \min_{X,U} \quad & \hat{J}_N(x_N) + \sum_{i=0}^{N-1} g_i(x_i, u_i) \\
\text{s.t.} \quad & x_{i+1} = f_i(x_i, u_i, w_i), \quad \forall i \in \{0, 1, 2, ...\} \\
& x_i \in \mathcal{X}, \quad \forall i \in \{0, 1, 2, ...\} \\
& u_i \in \mathcal{U}, \quad \forall i \in \{0, 1, 2, ...\} \\
& x_0 = \bar{x}_0 \\
& x_N \in \mathcal{X}_f
\end{aligned}
\right\} \quad \forall w_i \in \mathcal{W}, \forall i \in \{0, 1, 2, ...\}
$$

- We will use our Robust Invariant Set definitions and the Minkowski operations to change this problem into a suitable equivalent form.

Invariant Set Computation

- As we did before, suppose we solve the unconstrained infinite-horizon LQR problem:

$$
\begin{aligned}
J_0(\bar{x}_0) = \min_{X,U} \quad & \sum_{i=0}^{\infty} x_i^\top Q x_i + u_i^\top R u_i \\
\text{s.t.} \quad & x_{i+1} = Ax_i + Bu_i + w_i, \qquad \forall i \in \{0, 1, 2, \ldots\} \\
& x_0 = \bar{x}_0
\end{aligned}
$$

- Recall that in LQR, with have *certainty equivalence*. And the optimal closed-loop policy, is still given by the Ricatti Equation:

$$x_{k+1} = (A + BK)x_k + w_k$$

$$u_k^* = Kx_k$$

$$
\begin{cases}
K = -(B^\top K B + R)^{-1} B^\top P A \\
P = A^\top (P - PB(B^\top P B + R)^{-1} B^\top P) A + Q
\end{cases}
$$

Invariant Set Computation

- Now suppose we use our Invariant Set Algorithm to find the Robust Invariant Set w.r.t. the closed-loop system:

$$x_{k+1} = (A + BK)x_k + w_k$$

- Subject to the constraints:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : Hx \leq h \right\}, \forall k \geq 0$$

$$Kx_k \in \mathcal{U} = \left\{ x \in \mathbb{R}^n : H_u K x \leq h_u \right\}, \forall k \geq 0$$

$$w_k \in \mathcal{W} = \left\{ w \in \mathbb{R}^n : H_w w \leq h_w \right\}, \forall k \geq 0$$

Robust MPC

- Let $\mathcal{X}_f$ be this robust invariant set.
  - Note that it can be empty. Suppose it is not.

- Now let's focus on the first two stages:

$$x_1 = (A + BK)x_0 + w_0$$

$$x_2 = (A + BK)x_1 + w_1 = (A + BK)^2 x_0 + (A + BK)w_0 + w_1$$

- Then for a state k we can write:

$$x_k = (A + BK)x_{k-1} + w_{k-1} = (A + BK)^k x_0 + \sum_{i=0}^{k-1} (A + BK)^{k-1-i} w_i$$

Robust MPC

- Let's define a **nominal** state:

$$\bar{x}_k = (A + BK)^k x_0$$

- Which is the state value, if we had no disturbance.

- Now, observe that for state x_1 to be feasible we need to ensure that:

$$\bar{x}_1 \in \mathcal{X} \ominus \mathcal{W}$$

$$\mathcal{X} \ominus \mathcal{W} = \{x \in \mathbb{R}^n : x + w \in \mathcal{X}, \forall w \in \mathcal{W}\}$$

- Since:

$$x_1 = (A + BK)x_0 + w_0 = \bar{x}_1 + w_0$$

Robust MPC

- We can extend this to any stage k, by:

$$x_k = (A+BK)^k x_0 + \sum_{i=0}^{k-1}(A+BK)^{k-1-i}w_i = \bar{x}_k + \sum_{i=0}^{k-1}(A+BK)^{k-1-i}w_i$$

- So we need to ensure that:

$$\bar{x}_k \in \mathcal{X} \ominus \mathcal{R}_k$$

- Where:

$$\mathcal{R}_k = \oplus_{j=0}^{k-1}(A+BK)^j \circ \mathcal{W}$$

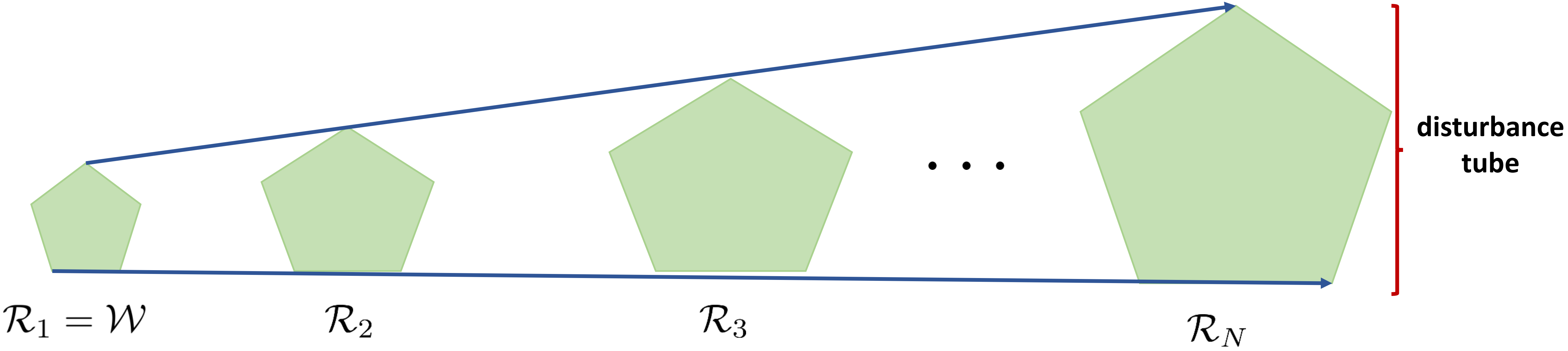
- Or in a recursive fashion:

$$\mathcal{R}_k = \mathcal{R}_{k-1} \oplus \left((A+BK)^{k-1} \circ \mathcal{W}\right) \qquad \mathcal{R}_0 = \{0\}$$

Robust MPC

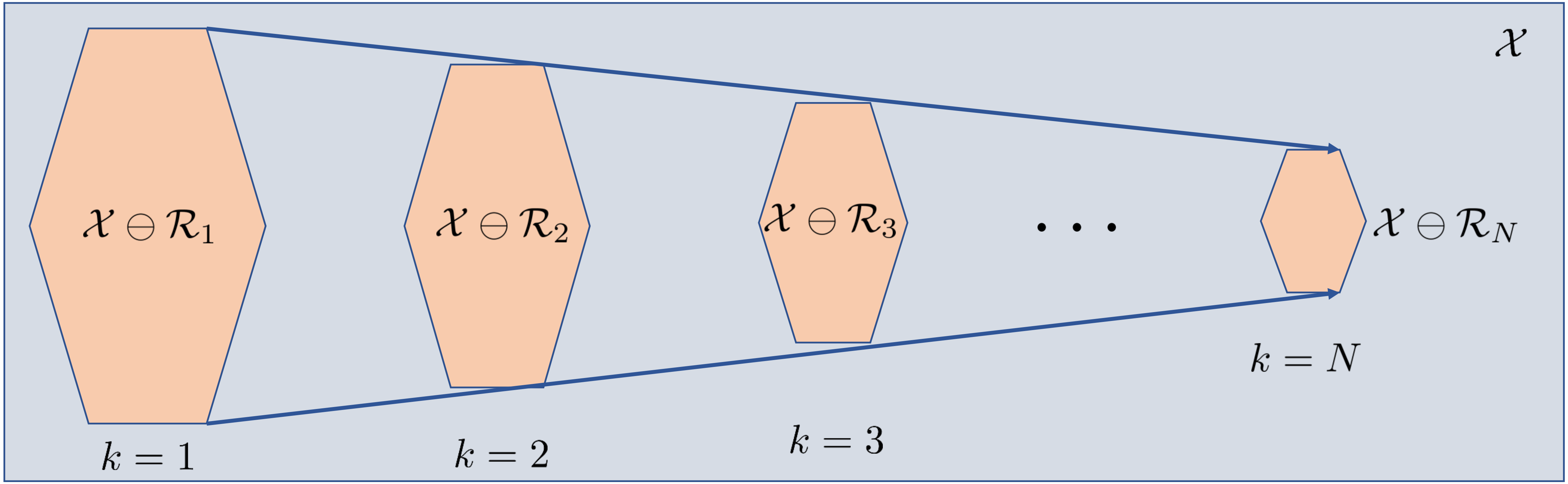
- The sequence $\{\mathcal{R}_k\}_{k=0}^{N-1}$ is often called the **disturbance tube**, because it represents how the disturbances can evolve until stage k.

- We can represent theses sets with a picture:



Robust MPC

- Then we can represent the evolution of the system as follows:



- As the disturbance tube get’s larger, the robust feasible region get’s smaller.

Robust MPC

- We can then write the Robust Optimal Control problem as follows:

$$J_0(\bar{x}_0) = \min_{X,U,Z} \quad \hat{J}_N(\bar{x}_N) + \sum_{i=0}^{N-1} \bar{x}_i^\top Q \bar{x}_i + u_i^\top R u_i$$

Nominal stage cost

$$\text{s.t.} \quad \bar{x}_{i+1} = A\bar{x}_i + Bu_i, \quad \forall i \in \{0, 1, 2, \dots\}$$

Nominal dynamics

$$\bar{x}_i \in \mathcal{X} \ominus \mathcal{R}_i, \quad \forall i \in \{1, 2, \dots\}$$

$$u_i = K\bar{x}_i + z_i, \quad \forall i \in \{0, 1, 2, \dots\}$$

z_i is effectively the control decision

$$u_i \in \mathcal{U} \ominus (K \circ \mathcal{R}_k), \quad \forall i \in \{0, 1, 2, \dots\}$$

$$x_0 = \bar{x}_0$$

$$x_N \in \mathcal{X}_f \ominus \mathcal{R}_N$$

Robust Invariant Set

Constraint Robustification

Robust MPC

- The Robust MPC Algorithm proceeds as before:

- We Solve the Robust Optimal Control problem.

- We apply the first stage control, and discard the rest.

- The **closed-loop** Robust MPC policy is given as:

$$\mu_{\mathrm{MPC}}(x_0) = u_0^*$$

- Then the system evolves to:

$$x_1 = Ax_0 + Bu_0^* + w_0$$

- We roll the horizon forward, and we repeat, now starting from x_1.

Robust MPC Properties

- All it remains is to answer the same two question as before. We need to ensure:
  - **(1) Recursive Feasibility**
  - **(2) (Robust) Asymptotic Stability**

- Recall that $\mathcal{X}_f$ is a robust control invariant set, associated with the LQR control law.

- Our terminal cost function approximation will be given as before:

$$\hat{J}_N(\bar{x}_N) = \bar{x}_N^\top P \bar{x}_N$$

- Lastly let $\mathcal{X}_0$ be the set of points such that the Optimal Control Problem is feasible.

- And we start from some state $x_0 \in \mathcal{X}_0$

Establishing Robust MPC Properties

- As we did before, let's start by proving feasibility. We start from a point $x_0 \in \mathcal{X}_0$

- So for the very first time step, the Optimal Control Problem is feasible with **nominal solution**:

$$(\bar{x}_0, u_0^*, \bar{x}_1, u_1^*, ..., \bar{x}_{N-1}, u_{N-1}^*, \bar{x}_N)$$

- Where:

$$u_k^* = K\bar{x}_k + z_k^*$$

- We apply the first-stage control u_0^* and discards the rest. The system evolves to:

$$x_1 = Ax_0 + Bu_0^* + w_0$$

- Now at x_1 consider the following control sequence: $(u_1^*, ..., u_{N-1}^*, K\bar{x}_N)$

Establishing Robust MPC Properties

- Now what we need to show is that after shifting the horizon forward the new nominal state $x'_k, k \geq 1$ is feasible. That is we need to show that:

$$x'_k \in \mathcal{X} \ominus \mathcal{R}_{k-1}$$

- Note that the "absolute" stage k, after we shift the horizon forward, becomes $k-1$
  - So stage 1 is now stage 0
  - Stage 2 is now stage 1
- Note that due to our constraint robustification it follows that:

$$x'_k = \bar{x}_k + (A+BK)^{k-1} w_0 \in (A\bar{x}_{k-1} + Bu^*_{k-1}) \oplus ((A+BK)^{k-1} \circ \mathcal{W}), \quad \forall k \geq 1$$

- And:

$$\bar{x}_k = A\bar{x}_{k-1} + Bu^*_{k-1} \in \mathcal{X} \ominus \mathcal{R}_k, \quad k \geq 1$$

Establishing Robust MPC Properties

- So we can write:

$$x'_k \in (A\bar{x}_{k-1} + Bu^*_{k-1}) \oplus ((A+BK)^{k-1} \circ \mathcal{W}) \subseteq \mathcal{X} \ominus \mathcal{R}_k \oplus ((A+BK)^{k-1} \circ \mathcal{W})$$

- By using the fact that:

$$\mathcal{R}_k = \mathcal{R}_{k-1} \oplus ((A+BK)^{k-1} \circ \mathcal{W})$$

- We have that:

$$\begin{aligned} x'_k &\in \mathcal{X} \ominus (\mathcal{R}_{k-1} \oplus ((A+BK)^{k-1} \circ \mathcal{W})) \oplus ((A+BK)^{k-1} \circ \mathcal{W}) \\ &\subseteq \mathcal{X} \ominus \mathcal{R}_{k-1} \\ &\Rightarrow x'_k \in \mathcal{X} \ominus \mathcal{R}_{k-1} \end{aligned}$$

Establishing Robust MPC Properties

- Now we can focus on the terminal state. The previous argument let's us write:

$$x'_N \in \mathcal{X}_f \ominus \mathcal{R}_{N-1}$$

- Now since $\mathcal{X}_f$ is robust control invariant, then it follows that:

$$x'_N \in ((A+BK)\circ\mathcal{X}_f)\ominus((A+BK)\circ\mathcal{R}_{N-1}) \subseteq (\mathcal{X}_f\ominus\mathcal{W})\ominus((A+BK)\circ\mathcal{R}_{N-1}) = \mathcal{X}_f\ominus\mathcal{R}_N$$

$$Kx'_N \in (K \circ \mathcal{X}_f) \ominus (K \circ \mathcal{R}_{N-1}) \subseteq \mathcal{U} \ominus (K \circ \mathcal{R}_{N-1})$$

- And lastly, for the very first state:

$$x_1 = \bar{x}_1 + w_0 \in (\mathcal{X} - \mathcal{W}) \oplus \mathcal{W} \subseteq \mathcal{X}$$

Establishing Robust MPC Properties

- Now arguing by Induction we can conclude the system is recursively feasible.

- Now let’s turn our attention to Asymptotically Stability.

- Note that because of the disturbance, our “goal” of driving the system to the origin is somewhat ill-posed.
  - We will “never” reach the origin as the disturbances will always cause perturbations in the system.

- Hence we need a new definition in order to capture this behavior.
  - We will extend our stability definition to Robust Asymptotically Stability.

Establishing Robust MPC Properties

- We will say that a system is robust asymptotically stable if:

- There exists a function $d(x,k) = \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}_+$:

- such that:
  - $d(0,k) = 0, \forall k$
  - $d(x,k)$ is continuous and strictly increases with $||x||$.
  - $d(\cdot,k)$ is decreasing and $d(x,k) \to 0$, for a fixed x, as $k \to \infty$

- And for each $\epsilon > 0$, $\exists \delta > 0$, such that for all disturbance values w_k satisfying:

$$w_k : \max_{k \geq 0} ||w_k|| < \delta$$

- We have that:

$$x_k \in \mathcal{X}, ||x_k|| \leq d(x_0,k) + \epsilon, \quad \forall k \geq 0$$

Establishing Robust MPC Properties

- This definition may seem non-intuitive

- A good intuitive explanation is to see the function $d(x,k)$ as a **distance function**, where it decreases as the system evolves (so k grows large)

- This distance is computed regarding some region around the origin.

- Then if the system is robust asymptotically stable then the system will be confined to a region around the origin.

- The size of the region depends on the disturbance magnitudes $||w_k||$
  - This is made explicit by using ϵ and δ in the previous definition

Establishing Robust MPC Properties

- In practice, this property can also be established by computing the distances to some smaller robust invariant set:

$$\mathcal{X}^* \subseteq \mathcal{X}_f \subseteq \mathcal{X}$$

- And the goal is to drive the system to $\mathcal{X}^*$.

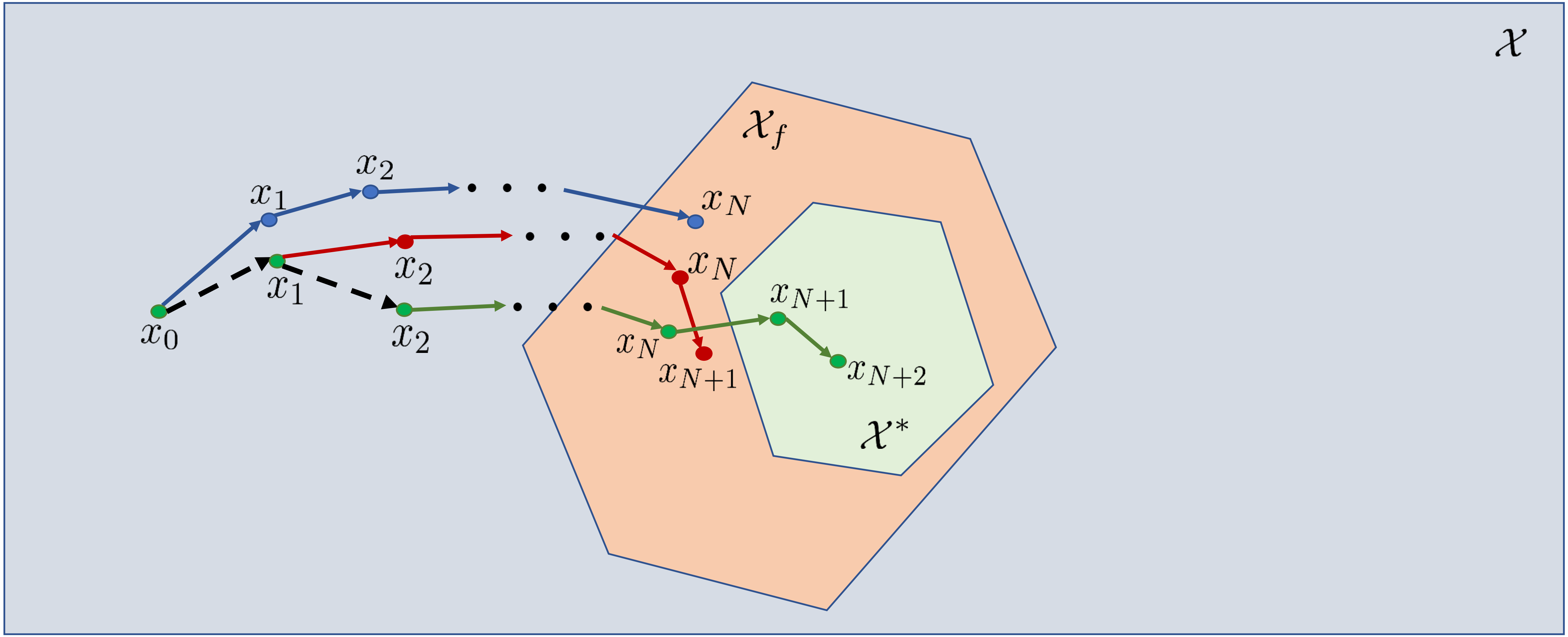
- It turns out that if the terminal cost approximation is given by:

$$\hat{J}_N(\bar{x}_N) = \bar{x}_N^\top P \bar{x}_N$$

- Then the Robust Linear MPC is Robust Asymptotically Stable.
  - The proof is a bit technical. We refer to "Model Predictive Control: Theory and Design, Rawlings and Mayne. 2009".
  - There is a small invariant set around the origin which the system will be confined to.

Illustration of Robust MPC

- We can Illustrate the Robust MPC Algorithm with a figure:



Remarks about Robust MPC

- We observe that even though the Robust MPC provides a **closed-loop** policy, we are solving an **open-loop** optimal control problem at every stage.
  - Our nominal model computes open-loop trajectories
  - We “close the loop” by implementing the first control and discarding the rest

- We can solve the Optimal Control Problem at every stage in closed-loop:
  - Via Stochastic Optimization
  - But it is very costly and hard general
  - If we do so, we say the Robust MPC has *closed-loop predictions*.

- Robust MPC is **very conservative** in practice:
  - We are “protecting” the system against all disturbances, even those that are not likely to occur.
  - The models needs to add a “learner” in order to reduce it’s conservatism.