Fractional Control Protocols in Linearized Relative Orbit Dynamics

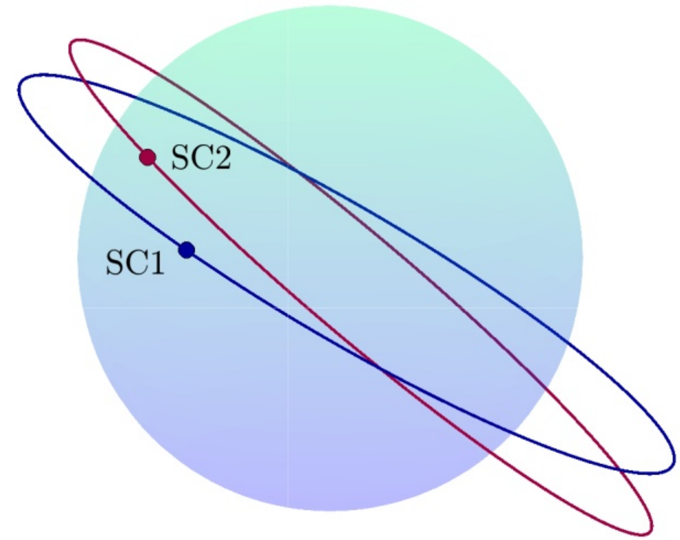
David Yaylali
Coauthored by:
Eric Butcher (UArizona)
Andrew Sinclair (AFRL)

University of Arizona,
Department of Aerospace and Mechanical Engineering

Two spacecraft in orbit about a common body

The trajectory of one spacecraft relative to another is known as the **relative orbit trajectory**.

SC1 → "Chief"
SC2 → "Deputy"



Observe the motion of the deputy from the "pilot's seat" of the chief

Chief Frame of Reference
(LVLH Frame)

$$\mathscr{L}\{\boldsymbol{\epsilon}_{\text{chief}}\} \leftrightarrow \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Dynamics of the Deputy
(exact, nonlinear description)

$$\ddot{x} = 2\dot{f}\left(\dot{y} + y\frac{\dot{r}}{r}\right) + x\dot{f}^2 + \frac{\mu}{r^2} - \frac{\mu}{r_d^3}(r + x)$$
$$\ddot{y} = -2\dot{f}\left(\dot{x} + x\frac{\dot{r}}{r}\right) + y\dot{f}^2 - \frac{\mu}{r_d^3}y$$
$$\ddot{z} = -\frac{\mu}{r_d^3}z$$

When orbits are very similar ($\delta\boldsymbol{\varepsilon} \to 0$), and when chief orbit is circular, these dynamics **linearize** to the...

(Hill-)Clohessy-Wiltshire (HCW) Equations

$$
\begin{aligned}
\ddot{x} &= 2n\dot{y} + 3n^2 x + u_x \\
\ddot{y} &= -2n\dot{x} \qquad\qquad\;\; + u_y \\
\ddot{z} &= -n^2 z \qquad\qquad\;\;\; + u_z
\end{aligned}
$$

Control thrusts on the deputy

The HCW equations are good approximations for **rendezvous** or **close-proximity formation flying**.

Clohessy and Wiltshire, "*Terminal Guidance System for Satellite Rendezvous*," J. Aero. Sci., 1960

Goal: *Design a controller which efficiently achieves rendezvous, $\{x, y, z\} \to 0$*

(assuming continuous control available, e.g., ion/EM thrusters)

To design controllers, it is convenient to "scale out" the dimensionful quantities...

$$(\text{length}) \to \frac{(\text{length})}{r_c}, \quad (\text{time}) \to \frac{(\text{time})}{T_{\text{orb}}} \quad \Rightarrow \quad \text{Control gains} \sim \mathcal{O}(1)$$

Dimensionless CW Equations

$$\begin{aligned} u'' &= \quad 2v' + 3u + \upsilon_u \\ v'' &= -2u' + \upsilon_v \\ w'' &= -w + \upsilon_w \end{aligned}$$

(derivatives w.r.t. chief true anomaly)

...or, in state-space form,

$$\dot{\boldsymbol{\chi}} = \begin{bmatrix} \mathbf{0} & I_3 \\ A_1 & A_2 \end{bmatrix} \boldsymbol{\chi} + \begin{bmatrix} \mathbf{0} \\ I_3 \end{bmatrix} \boldsymbol{\upsilon} \qquad A_1 = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}, \ A_2 = \begin{bmatrix} 0 & 2 & 0 \\ -2 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Design a **controller for the deputy** which alters its trajectory relative to the chief.

$$\dot{\boldsymbol{\chi}} = \begin{bmatrix} \mathbf{0} & I_3 \\ A_1 & A_2 \end{bmatrix} \boldsymbol{\chi} + \begin{bmatrix} \mathbf{0} \\ I_3 \end{bmatrix} \boldsymbol{\upsilon}$$

Full-State Feedback:

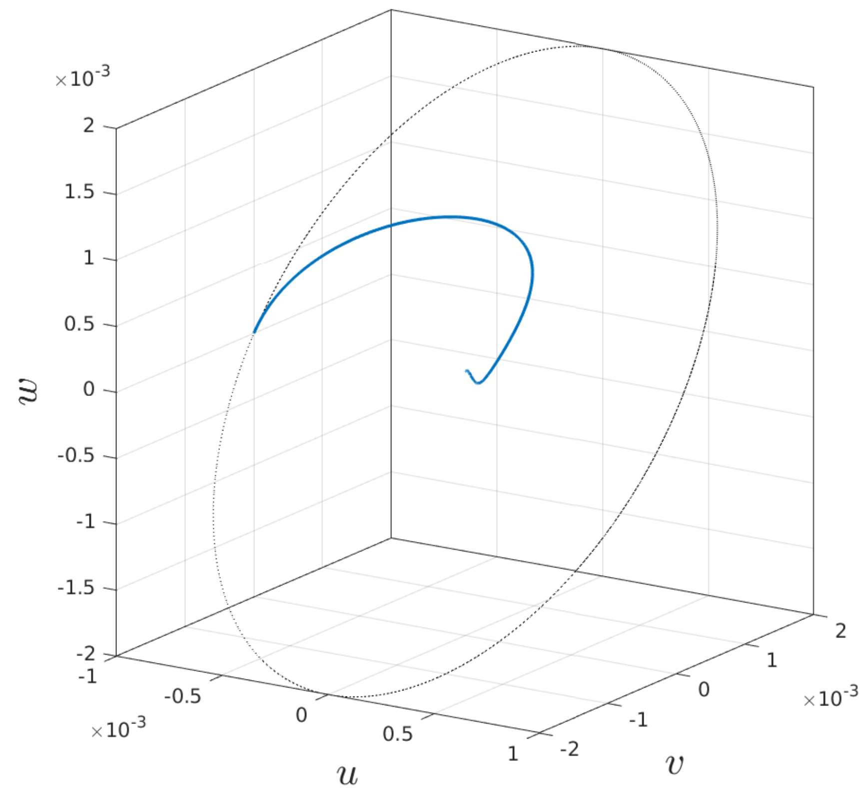
$$\boldsymbol{\upsilon} = -K\boldsymbol{\chi} = -\begin{bmatrix} \underset{(3\times 3)}{K_P} & \underset{(3\times 3)}{K_D} \end{bmatrix} \begin{bmatrix} \boldsymbol{\rho} \\ \dot{\boldsymbol{\rho}} \end{bmatrix}$$

This is a **proportional-derivative** (PD) controller: $\boldsymbol{\upsilon} = -K_P\boldsymbol{\rho} - K_D D^1 \boldsymbol{\rho}$

Example: LQR Design

$$\delta J = \delta \int_0^\infty \left(\boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{u}^T R \boldsymbol{u} \right) dt = 0$$

$$Q = I_4, \quad R = I_2 \quad \Rightarrow \quad K_{\mathrm{LQR}}$$



Design a **controller for the deputy** which alters its trajectory relative to the chief.

$$\dot{\boldsymbol{\chi}} = \begin{bmatrix} \mathbf{0} & I_3 \\ A_1 & A_2 \end{bmatrix} \boldsymbol{\chi} + \begin{bmatrix} \mathbf{0} \\ I_3 \end{bmatrix} \boldsymbol{\upsilon}$$

Full-State Feedback:

$$\boldsymbol{\upsilon} = -K\boldsymbol{\chi} = -\begin{bmatrix} \underset{(3\times 3)}{K_P} & \underset{(3\times 3)}{K_D} \end{bmatrix} \begin{bmatrix} \boldsymbol{\rho} \\ \dot{\boldsymbol{\rho}} \end{bmatrix}$$

This is a **proportional-derivative** (PD) controller: $\boldsymbol{\upsilon} = -K_P\boldsymbol{\rho} - K_D D^1 \boldsymbol{\rho}$

- Limited ability to adjust the controlled trajectory.
  Example: Reducing overshoot increases settling time and/or control effort

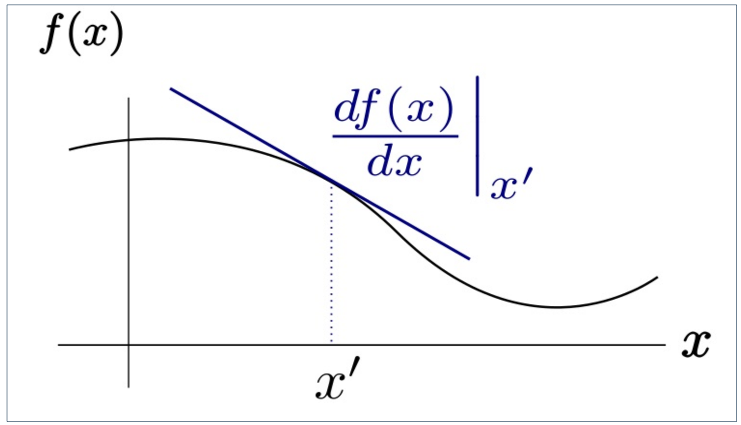
- Increase freedom in shaping the trajectory by utilizing ***fractional derivative control:***

$$\boldsymbol{\upsilon} = -K_P\boldsymbol{\rho} - K_D D^\alpha \boldsymbol{\rho}$$

Our main point: *We can achieve **more optimal** trajectories using fractional control*

Derivatives of common experience can only be “wholly” applied:

A “whole” derivative of $f(x)$



Similarly, we can apply derivatives any (natural) number of times:

$$\underbrace{\frac{d}{dx}\frac{d}{dx}\cdots\frac{d}{dx}}_{n \text{ times}} f(x) = D^n f(x)$$

Common derivative and integral operators are **integer-ordered:**

$$D^n, \quad n \in \mathbb{Z}$$

*Can we generalize to **real-ordered** derivatives?*

Derivatives of a power function:

$$\frac{d}{dx}(x^m) = m x^{m-1}$$
$$\frac{d^2}{dx^2}(x^m) = m(m-1)x^{m-2}$$
$$\vdots$$
$$\frac{d^n}{dx^n}(x^m) = m(m-1)\cdots(m-n+1)x^{m-n}$$

In general, this can be written

$$\frac{d^n}{dx^n}(x^m) = \frac{m!}{(m-n)!}x^{m-n}$$

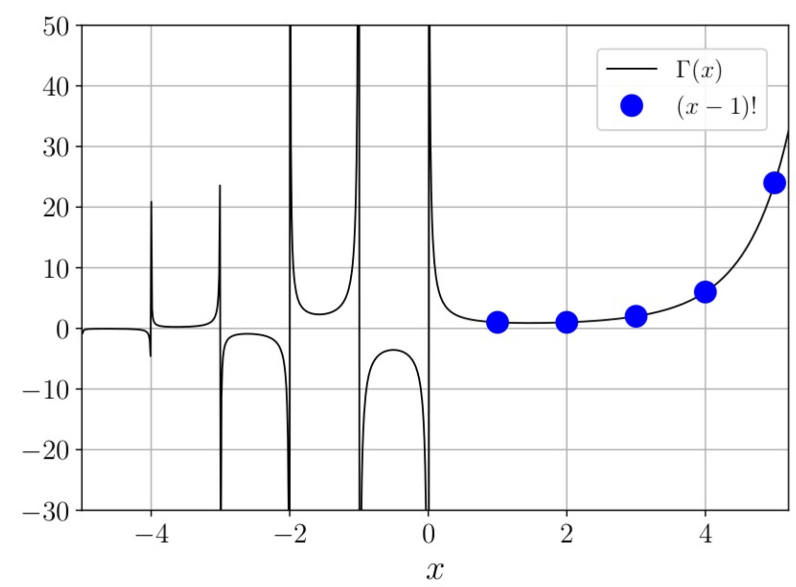
$$\text{for } m, n \in \mathbb{Z}, \quad m \geq n$$

Euler's *generalization* of the factorial function:

$$(n-1)! \quad \longrightarrow \quad \Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx = (z-1)!$$

Valid for all real numbers z

(excluding the negative integers)

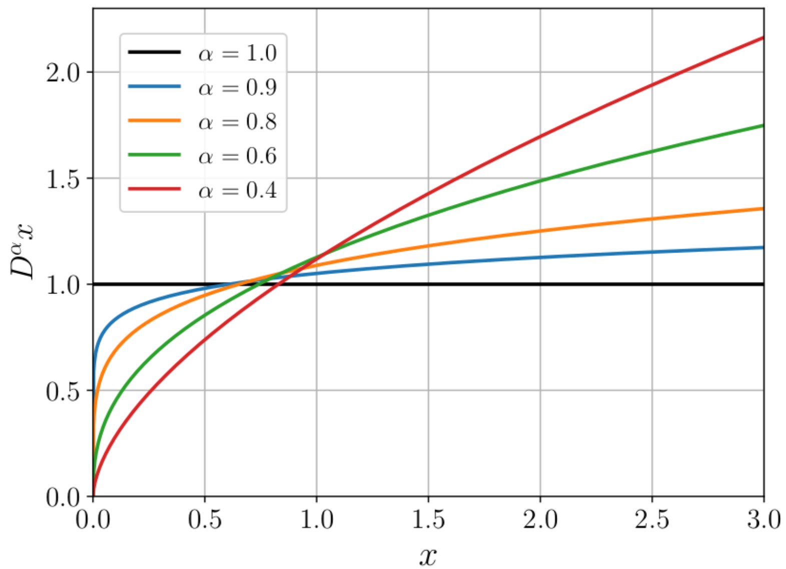


We can therefore *continue* the derivative of a power function to **any real order**!

"Fractional" derivative:

$$\frac{d^\alpha}{dx^\alpha}(x^m) = \frac{\Gamma(m+1)}{\Gamma(m-\alpha+1)} x^{m-\alpha}, \quad m \in \mathbb{Z},\ \alpha \in \mathbb{R}$$

Example: Fractional derivatives of x



e.g., half-derivative of x

$$\frac{d^{1/2}}{dx^{1/2}} x = \frac{\Gamma(2)}{\Gamma(3/2)} x^{1/2} = \frac{2}{\sqrt{\pi}} x^{1/2}$$

Details aside...

Fractional calculus → Fractional derivatives and integrals of **general functions**

Derivatives can now be "continued" to non-integer order

$$\frac{d^n f(x)}{dx^n}, \;\; n \in \mathbb{Z} \quad \Rightarrow \quad D^\alpha f(x) \equiv \frac{d^\alpha f(x)}{dx^\alpha}, \;\; \alpha \in \mathbb{R}$$

...which we can use to build fractional relative-orbit controllers:

$$\dot{\boldsymbol{\chi}} = \begin{bmatrix} \mathbf{0} & I_3 \\ A_1 & A_2 \end{bmatrix} \boldsymbol{\chi} + \begin{bmatrix} \mathbf{0} \\ I_3 \end{bmatrix} \boldsymbol{\upsilon}$$

$$\boldsymbol{\upsilon} = -K_P \boldsymbol{\rho} - K_D D^1 \boldsymbol{\rho} \quad \Rightarrow$$

Fractional PD-Type Controller

$$\boldsymbol{\upsilon} = -K_P \boldsymbol{\rho} - K_D D^{\boldsymbol{\alpha}} \boldsymbol{\rho}$$

The orders α are additional tunable control parameters
(and categorically different from the P and D gains)

Fractional Consensus Controller

Note this is no longer "full state feedback":

$$\boldsymbol{\upsilon} = -K_P\boldsymbol{\rho} - K_D D^{\boldsymbol{\alpha}}\boldsymbol{\rho} \neq -\begin{bmatrix}K_P & K_D\end{bmatrix}\begin{bmatrix}\boldsymbol{\rho}\\ \dot{\boldsymbol{\rho}}\end{bmatrix}$$

$$\dot{\boldsymbol{\chi}} = \begin{bmatrix}\mathbf{0} & I_3\\ A_1 & A_2\end{bmatrix}\boldsymbol{\chi} + \begin{bmatrix}\mathbf{0}\\ I_3\end{bmatrix}\boldsymbol{\upsilon} \quad \not\Leftrightarrow \quad \dot{\boldsymbol{\chi}} = \begin{bmatrix}\mathbf{0} & I_3\\ A_1 - K_P & A_2 - K_D\end{bmatrix}\boldsymbol{\chi}$$

So evolution/stability conditions are different from standard LTI systems

***Details aside**, we can recover analogous formalism by describing the system using a pseudostate,*

$$D^{\alpha}\boldsymbol{X} = \tilde{A}\boldsymbol{X} + \tilde{B}\boldsymbol{u}$$

$\Rightarrow$ *Pseudo*state transition matrix, stability based on eigenvalues, etc...

Proof-of-principle

Again, our main goal is to show that the additional tunable control parameter can give more optimal trajectories.

In other words, can fractional controllers give rendezvous trajectories that are…

Faster? ⟺ Settling time (ST)

More direct? ⟺ Overshoot (OS)

Cheaper? ⟺ Integrated Control Effort (U)
($\propto$ fuel cost)

All of the above???

These can be appropriately defined for relative orbit control-to-rendezvous.

Proof-of-principle

Again, our main goal is to show that the additional tunable control parameter can give more optimal trajectories.

In other words, can fractional controllers give rendezvous trajectories that are…

Faster?	$\Longleftrightarrow$	Settling time (ST)
More direct?	$\Longleftrightarrow$	Overshoot (OS)
Cheaper?	$\Longleftrightarrow$	Integrated Control Effort (U) ($\propto$ fuel cost)
All of the above???		

These can be appropriately defined for relative orbit control-to-rendezvous.

We therefore currently strive only for ***proof-of-principle* examples** where this can be achieved.

- We will consider three benchmark relative orbits
- We will optimize numerically

  This is technically not an “optimal control” problem… optimal fractional control is a topic for future work.

Benchmark initial conditions:

Chief orbit:
(Earth orbiting)

$$\epsilon_{\text{chief}} = \{a,\ e,\ i\ ,\Omega,\ \omega,\ \nu\} = \{10^4\ \text{km}, 0, 20^\circ, 30^\circ, 0^\circ, 0^\circ\}$$

	u	v	w	$\dot{u}$	$\dot{v}$	$\dot{w}$
IC1	-0.001	0	0	0	0.002	0.002
IC2	-0.001	0.002	-0.001	0	0.002	0
IC3	0.001	-0.002	0.001	0	0.002	-0.002

These correspond to approximately the following OE differences:

	δa	δe	δi	$\delta\Omega$	$\delta\omega$	$\delta\nu$
IC1	0	0.001	0.1°	0	0	0
IC2	0	0.001	0	0.17°	0	-0.05°
IC3	80 km	0.007	-0.1°	-0.17°	0	0.05°

Since the components in the CW dynamics decouple...

In-Plane Motion – *u* and *v* components:

$$u'' = 2v' + 3u + \upsilon_u$$
$$v'' = -2u' + \upsilon_v$$

Control: $\boldsymbol{\upsilon}_\xi = -K_P \begin{bmatrix} u \\ v \end{bmatrix} - K_D \begin{bmatrix} D^{\alpha_u} u \\ D^{\alpha_v} v \end{bmatrix}$

10 free controller parameters: $\{K_P, K_D, \alpha_u, \alpha_v\}$

Out-of-Plane Motion – *w* component:

$$w'' = -w + \upsilon_w$$

Control: $\upsilon_w = -k_{P_w} w - k_{D_w} D^{\alpha_w} w$

3 free controller parameters: $\{k_{P_w}, k_{D_w}, \alpha_w\}$

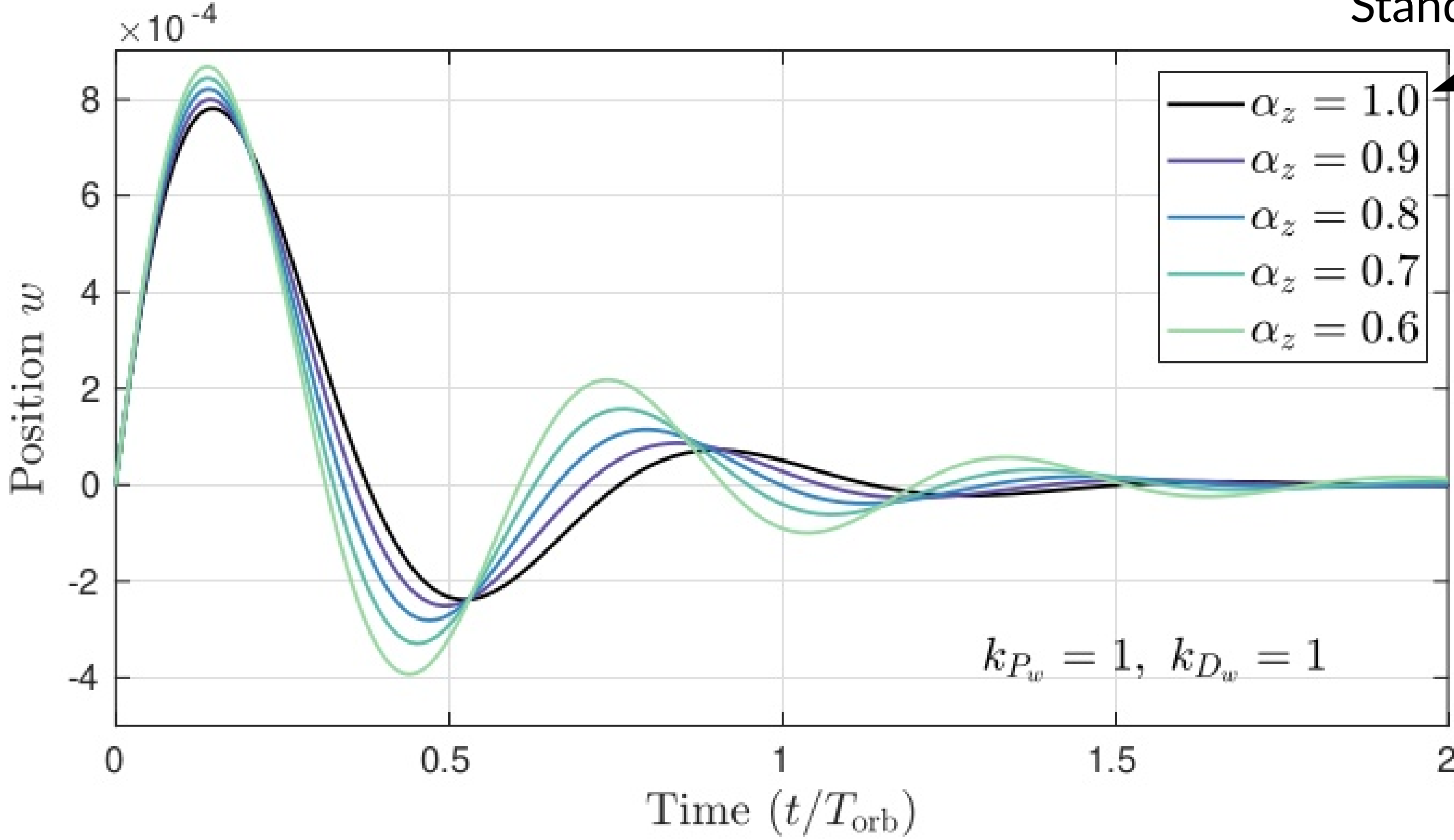
...we will consider these separately.

Out-of-Plane Motion

Fractional-control for out-of-plane motion essentially generalizes from a damped harmonic oscillator to a **fractionally damped harmonic oscillator**:

$$w'' + k_{D_w} w' + (1 + k_{P_w})w = 0 \quad \Rightarrow \quad w'' + k_{D_w} D^{\alpha_w} w + (1 + k_{P_w})w = 0$$

Standard controller



$$w'' + k_{D_w} D^{\alpha_w} w + (1 + k_{P_w}) w = 0$$

We can now survey over all free controller parameters,

$$\{k_{P_w}, k_{D_w}, \alpha_w\}$$

and optimize some performance measure ψ as a function of fractional order α

Strategy:

- For a given α_w, find gains which optimize a performance measure ψ:

$$\{k^*_{P_w}, k^*_{D_w}\} \Rightarrow w^*(t) \quad \Leftrightarrow \quad \text{Optimal gains in "}\alpha\text{" slice}$$

- Find ψ_{opt} for different slices, corresponding to different fractional-order.

$$\alpha^*_w \quad \Leftrightarrow \quad \text{Optimal "}\alpha\text{" slice}$$

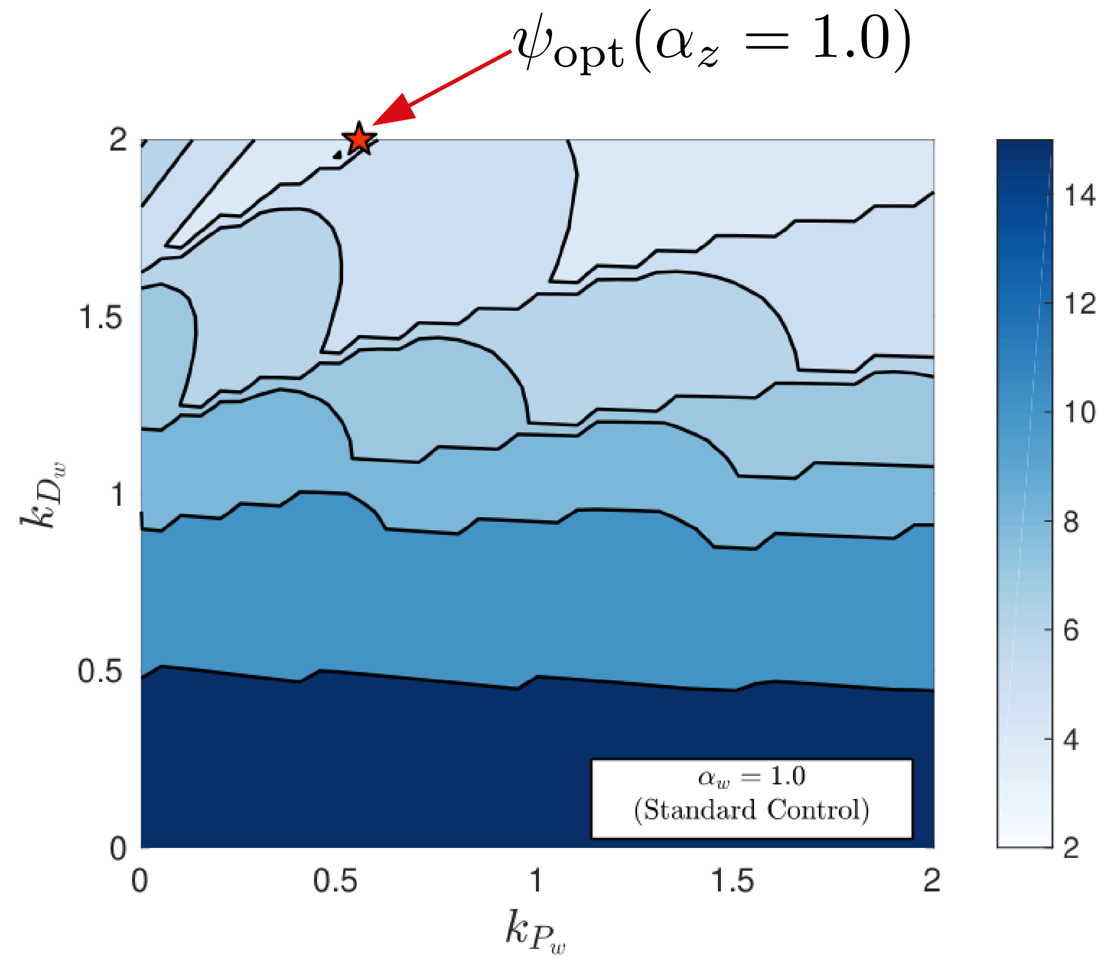
- If...

$$\psi_{\text{opt}}(\alpha^*_w) < \psi_{\text{opt}}(1)$$

...then fractional controller (of order α^*_w) **outperforms standard controller**.

Optimizing settling time: $\psi = \mathrm{ST}$

Considering IC1...

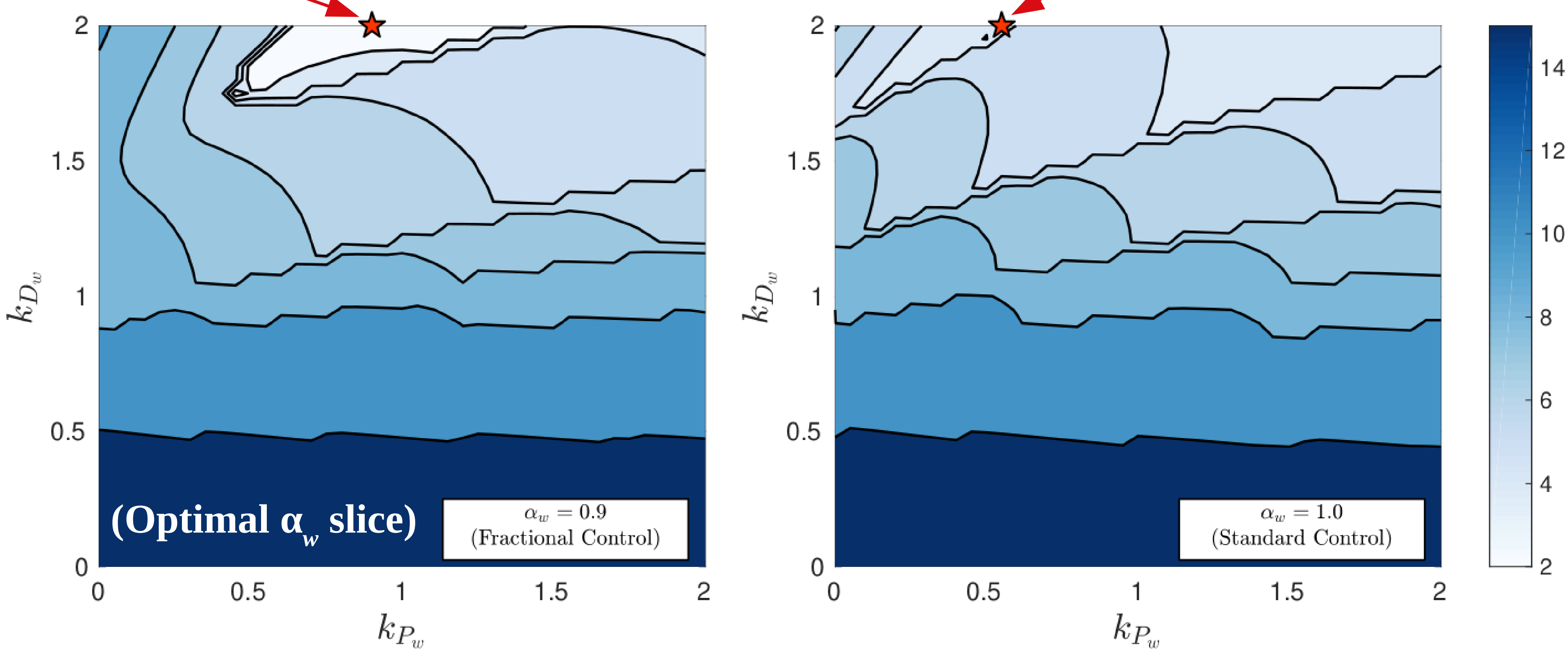


Optimizing settling time: $\psi = \mathrm{ST}$

Considering IC1…

$\psi_{\text{opt}}(\alpha_w = 0.91)$

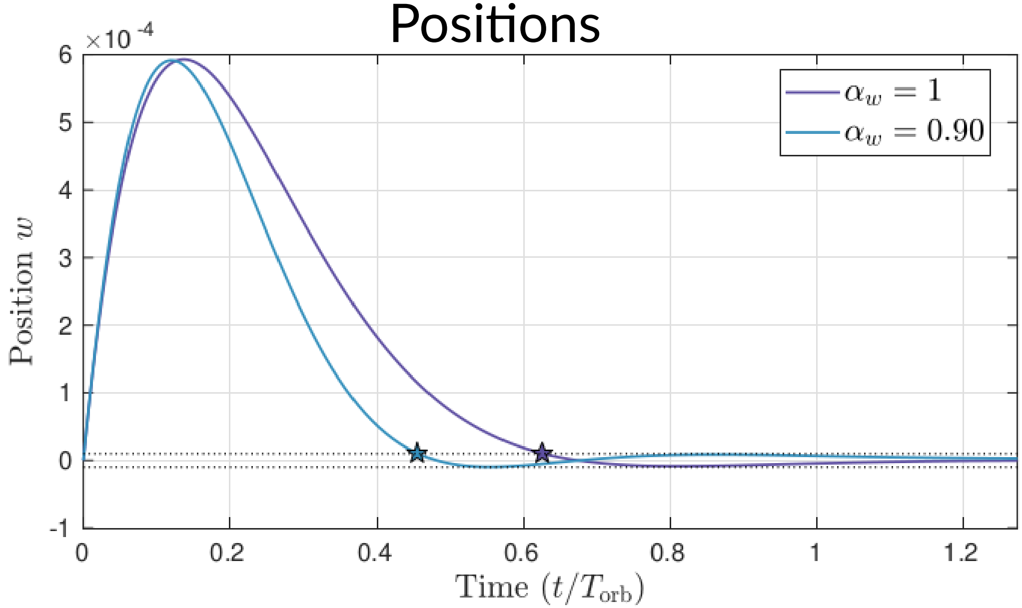
$\psi_{\text{opt}}(\alpha_z = 1.0)$



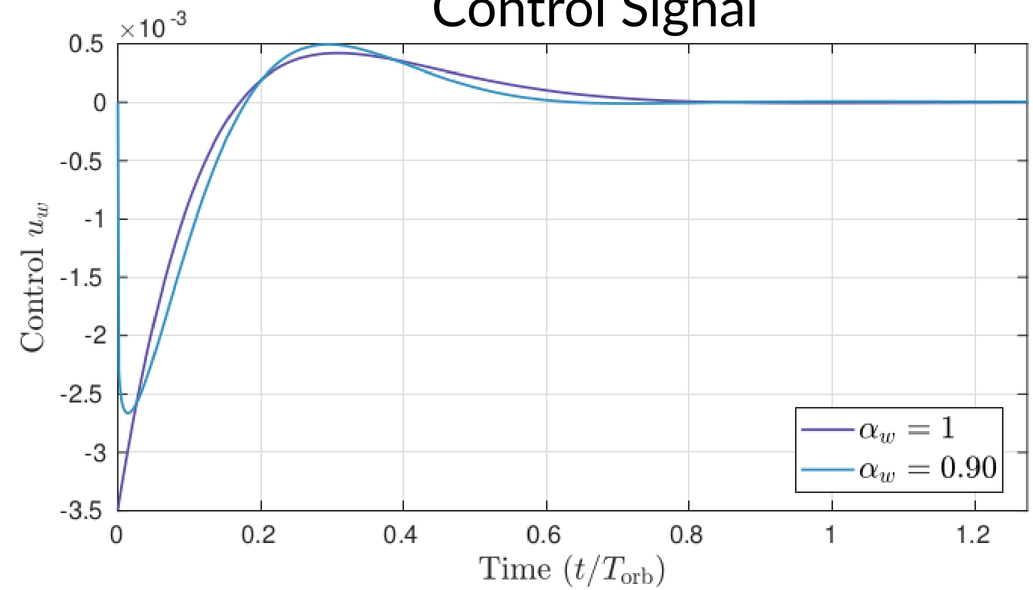
Now compare trajectories for these two optimal parameter choices

Optimized Trajectories:

Positions



Control Signal



Settling Time:

$$\tau_{\rm int} = 0.625 T_{\rm orb}$$
$$\tau_{\rm frac} = 0.455 T_{\rm orb}$$

27% reduction in settling time…

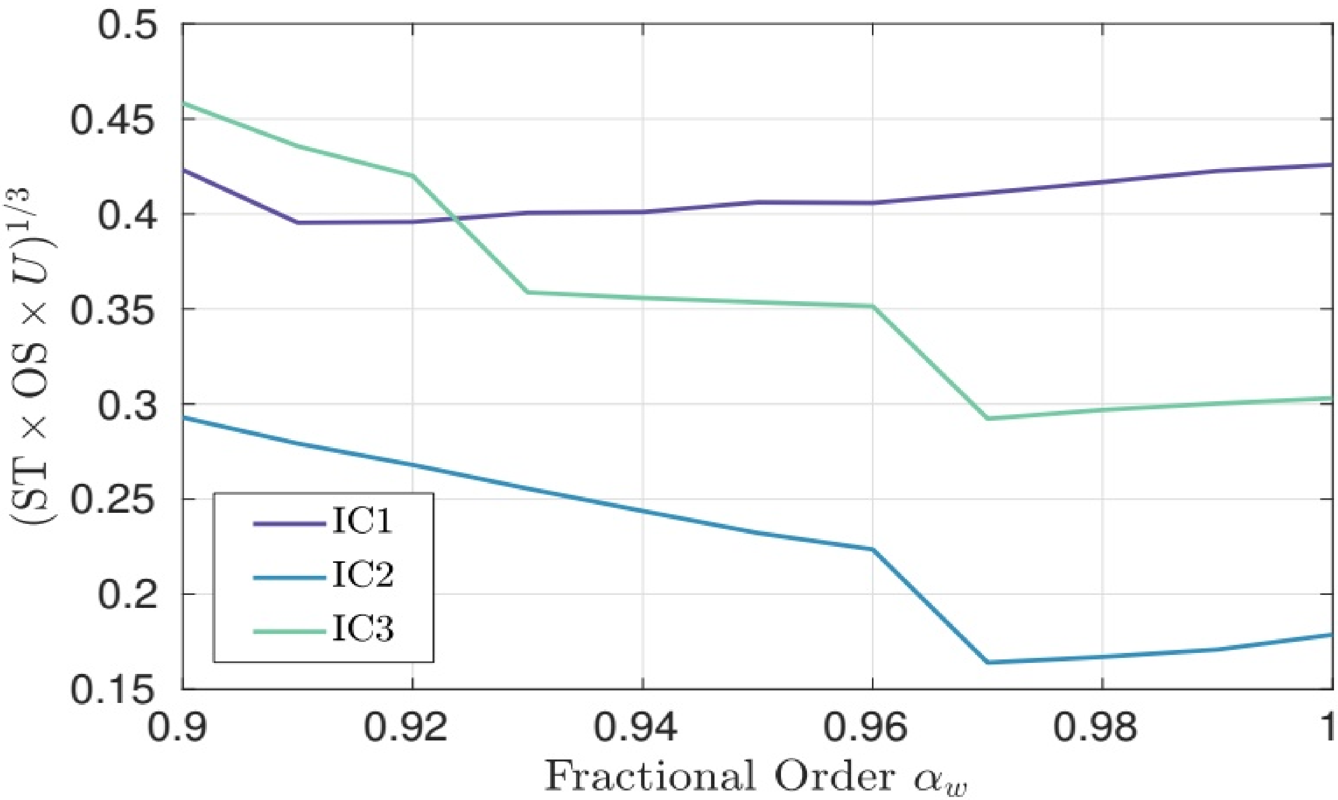
Integrated Control:
(fuel cost)

$$U_{\rm int} = 0.00242$$
$$U_{\rm frac} = 0.00232$$

…with less fuel spent

Optimizing ST, OS, and U simultaneously: $\psi = (\mathrm{ST} \times \mathrm{OS} \times U)^{1/3}$

Considering all benchmark ICs...



In all three cases, fractional control gives more optimal performance

In-Plane Motion

Fractional-control for in-plane motion generally corresponds to two coupled **fractionally damped harmonic oscillators.**

$$\boldsymbol{v}_\xi = -K_P \begin{bmatrix} u \\ v \end{bmatrix} - K_D \begin{bmatrix} D^{\alpha_u} u \\ D^{\alpha_v} v \end{bmatrix}$$

10 design parameters:

$$\begin{bmatrix} K_P & K_D \end{bmatrix} = \begin{bmatrix} k_{p_{11}} & k_{p_{12}} & k_{d_{11}} & k_{d_{12}} \\ k_{p_{21}} & k_{p_{22}} & k_{d_{21}} & k_{d_{22}} \end{bmatrix} \qquad \boldsymbol{\alpha} = \begin{bmatrix} \alpha_u \\ \alpha_v \end{bmatrix}$$

We again want to survey over controller parameters in order to optimize some performance measure…

A standard grid-based survey over this gain space **obviously will not work**:

Computation time:
(for one grid point on my desktop PC)

$$\approx 0.1 \text{ sec}$$

Desired gain grid-resolution:
(i.e., gain precision)

$$\delta k = 0.05$$

Number of points in survey grid:
(if varying gain between -5 and 5)

$$2.56 \times 10^{18}$$

Time needed for naive brute-force survey:

$$t_{\text{comp}} \approx 2 \times 10^{17} \text{ sec} \approx \frac{1}{2}\, t_{\text{universe}}$$

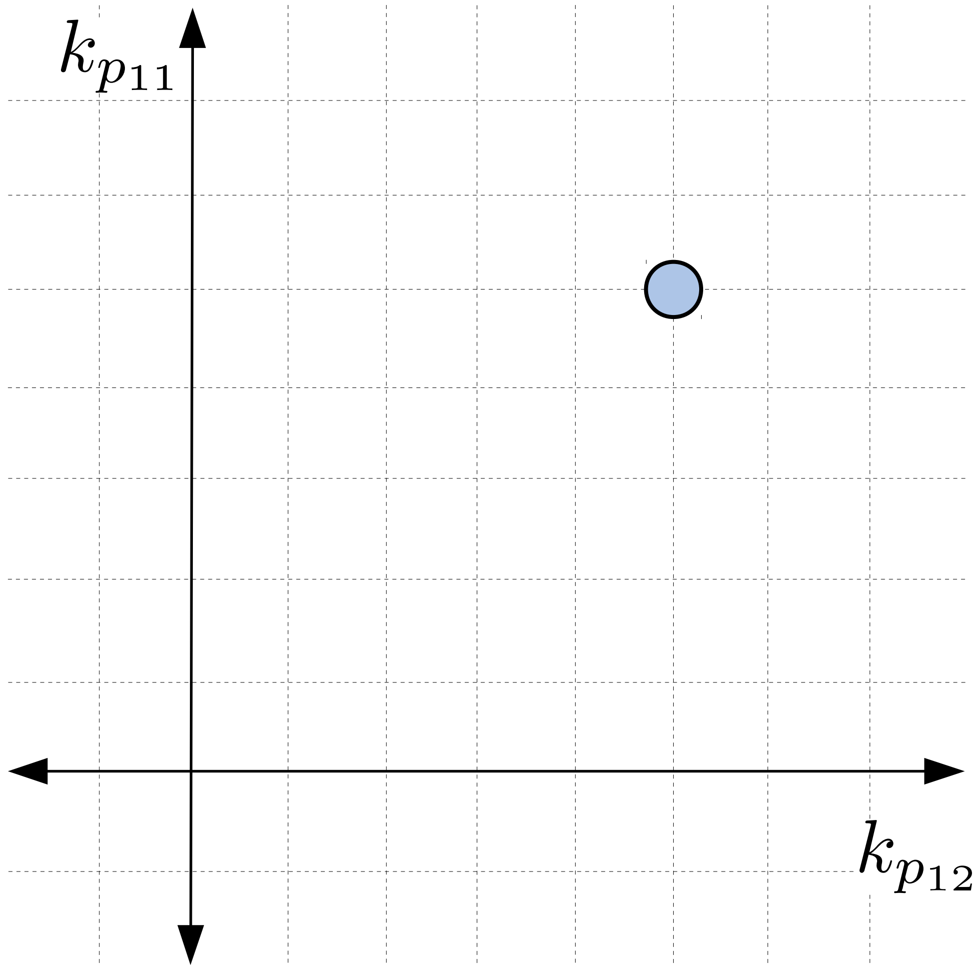
Need optimization search algorithms

- Ideally we need a global (non-convex) optimization algorithm.
- We seek *proof-of-principle examples* → use a simple **pattern search algorithm** to find the closest **local minimum** to some starting point.

Basic Pattern Search Algorithm:

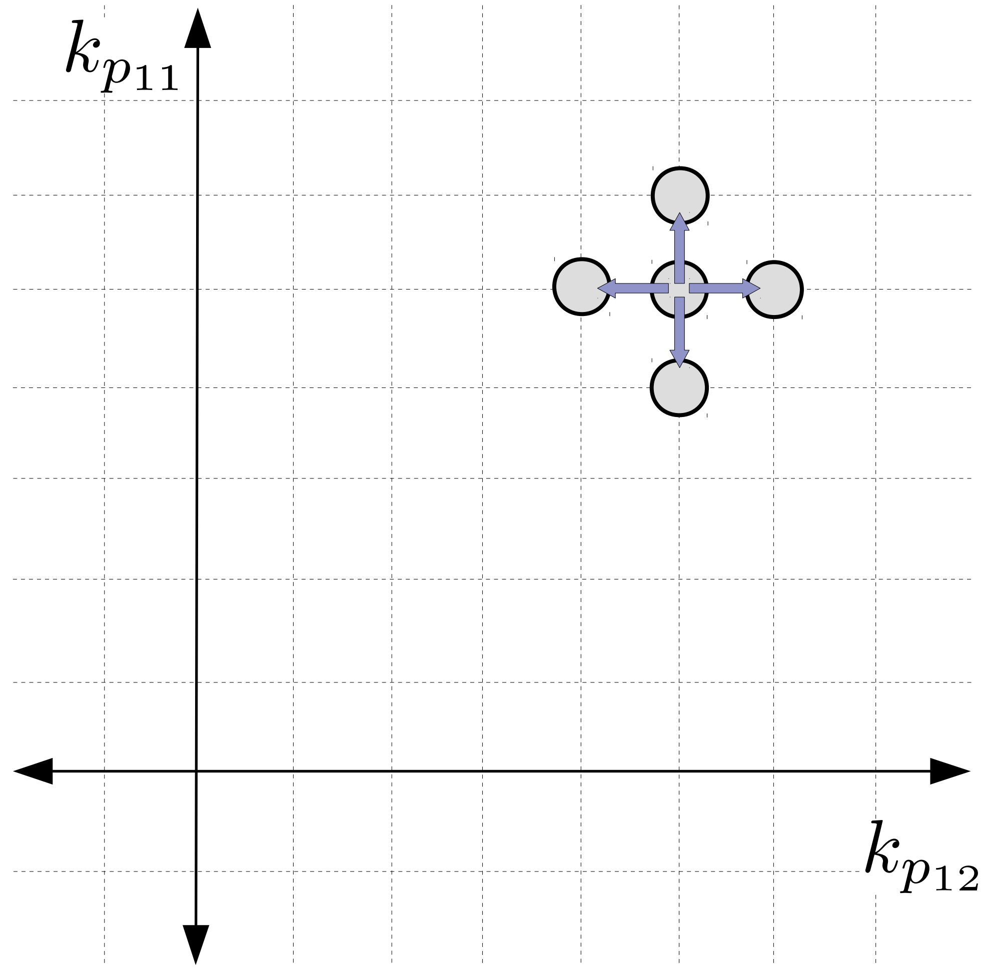
- Compute performance measure at starting point.

We choose LQR as our starting point, since this will give reasonably small values of ST, OS, and U



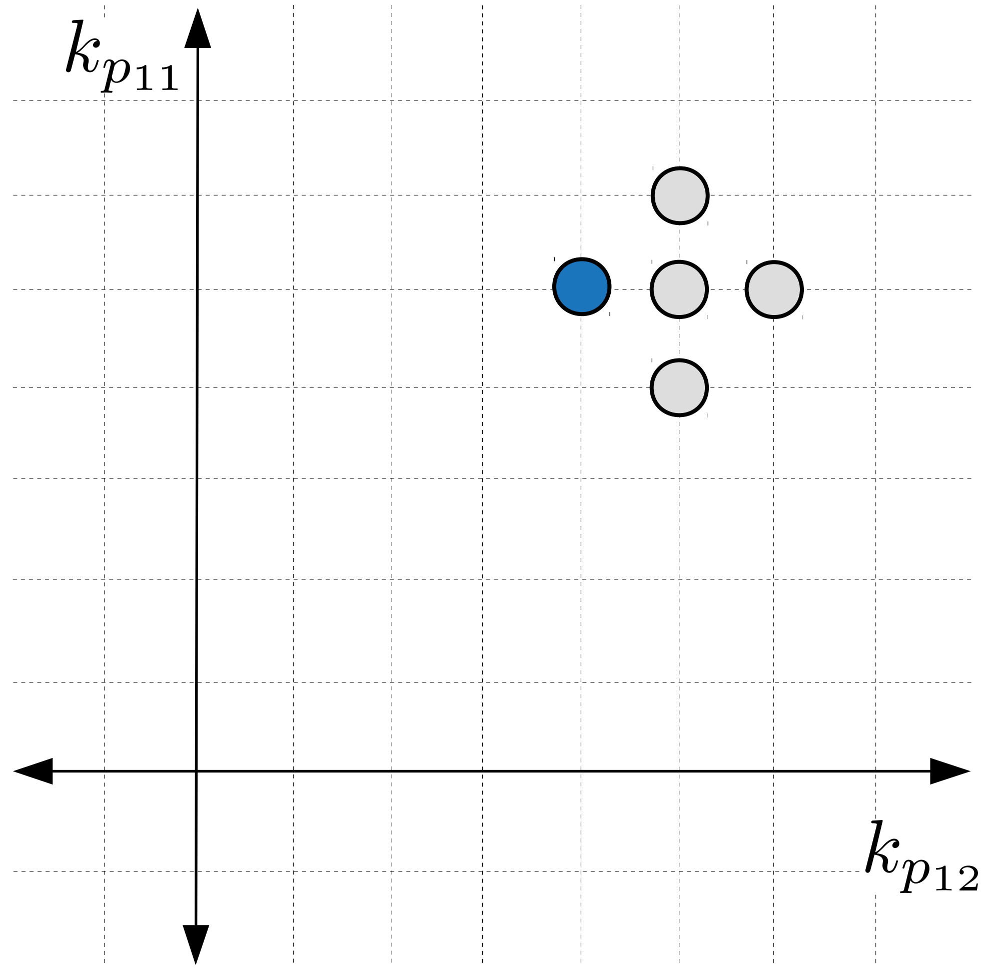
Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points
  (16 points in full 8D gain space)



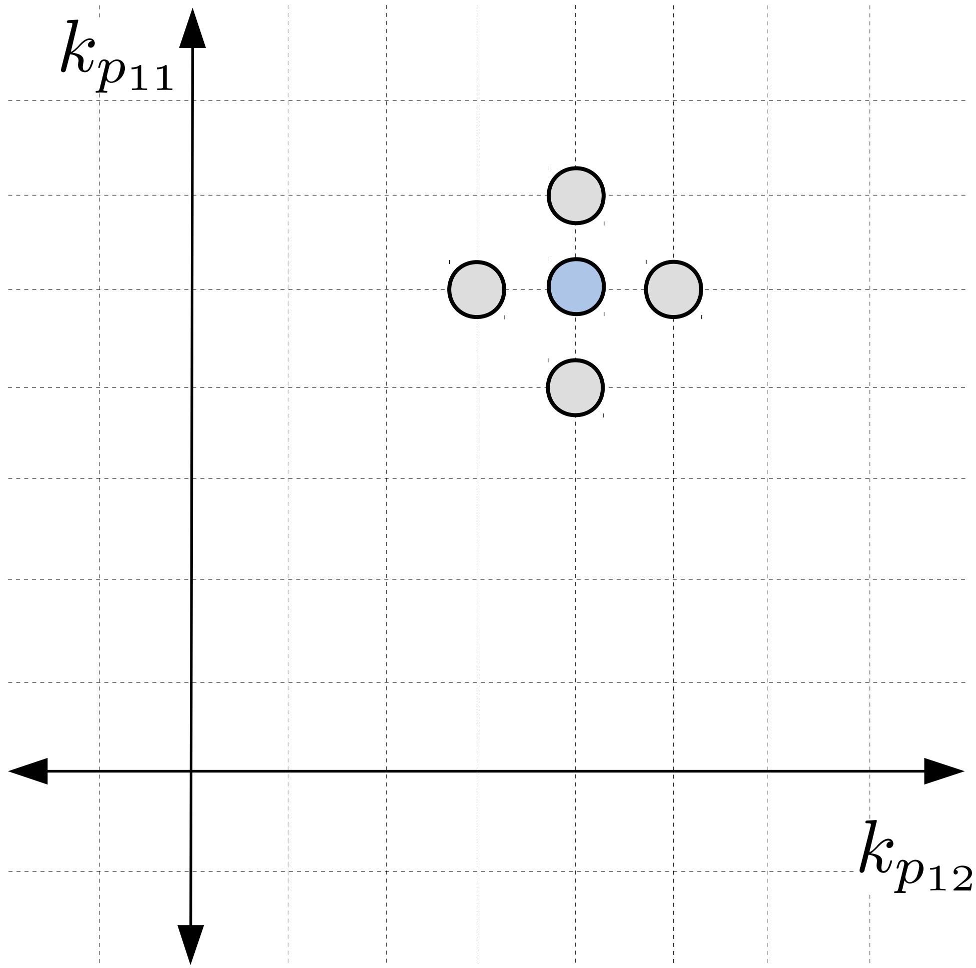
Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points
  (16 points in full 8D gain space)
- Find minimum performance measure.



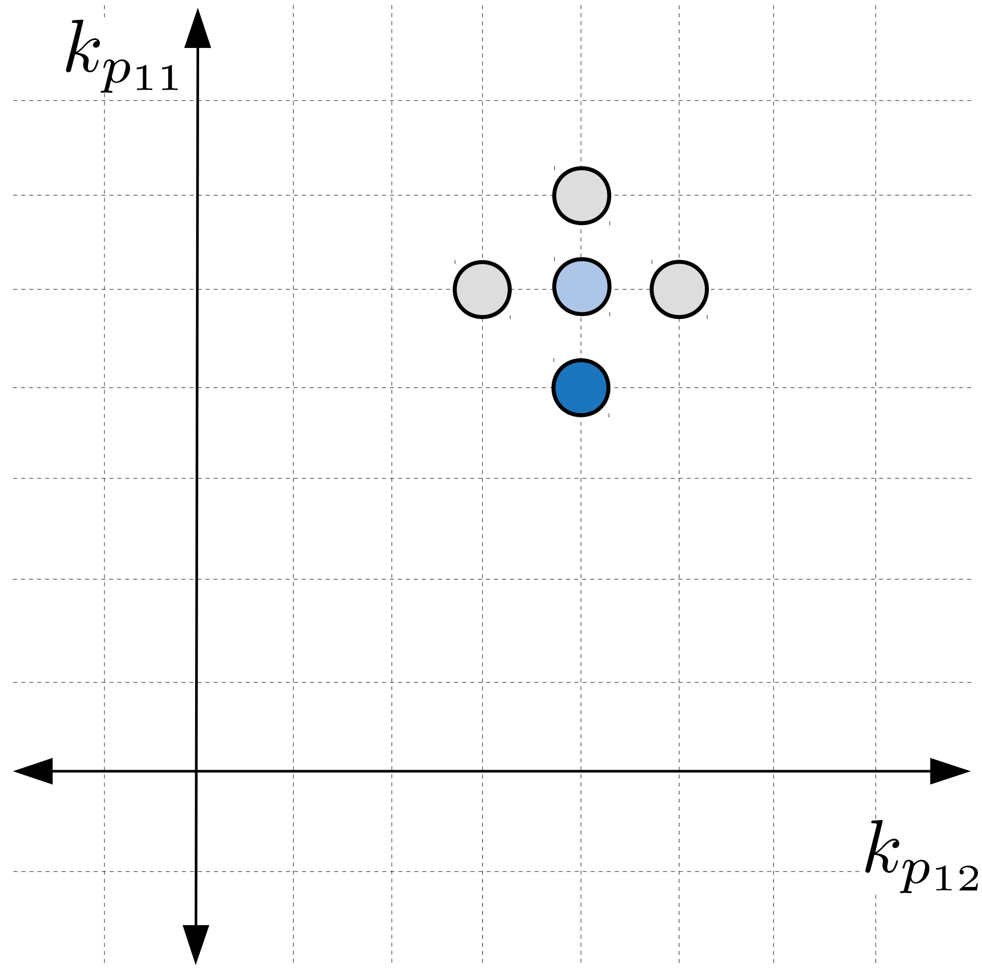
Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points
  (16 points in full 8D gain space)
- Find minimum performance measure.
- Move to new minimum, **then recompute** performance measure at all neighboring points.



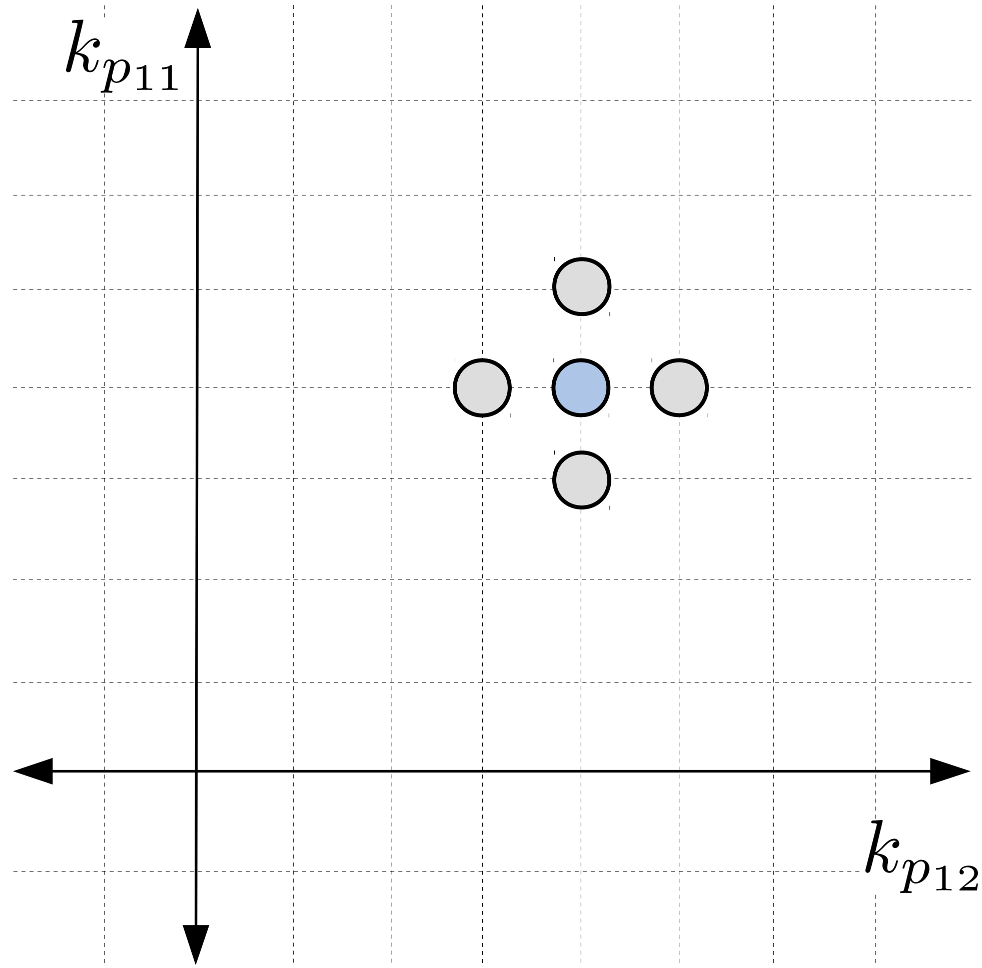
Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points
  (16 points in full 8D gain space)
- Find minimum performance measure.
- Move to new minimum, **then recompute** performance measure at all neighboring points.
- Repeat algorithm...



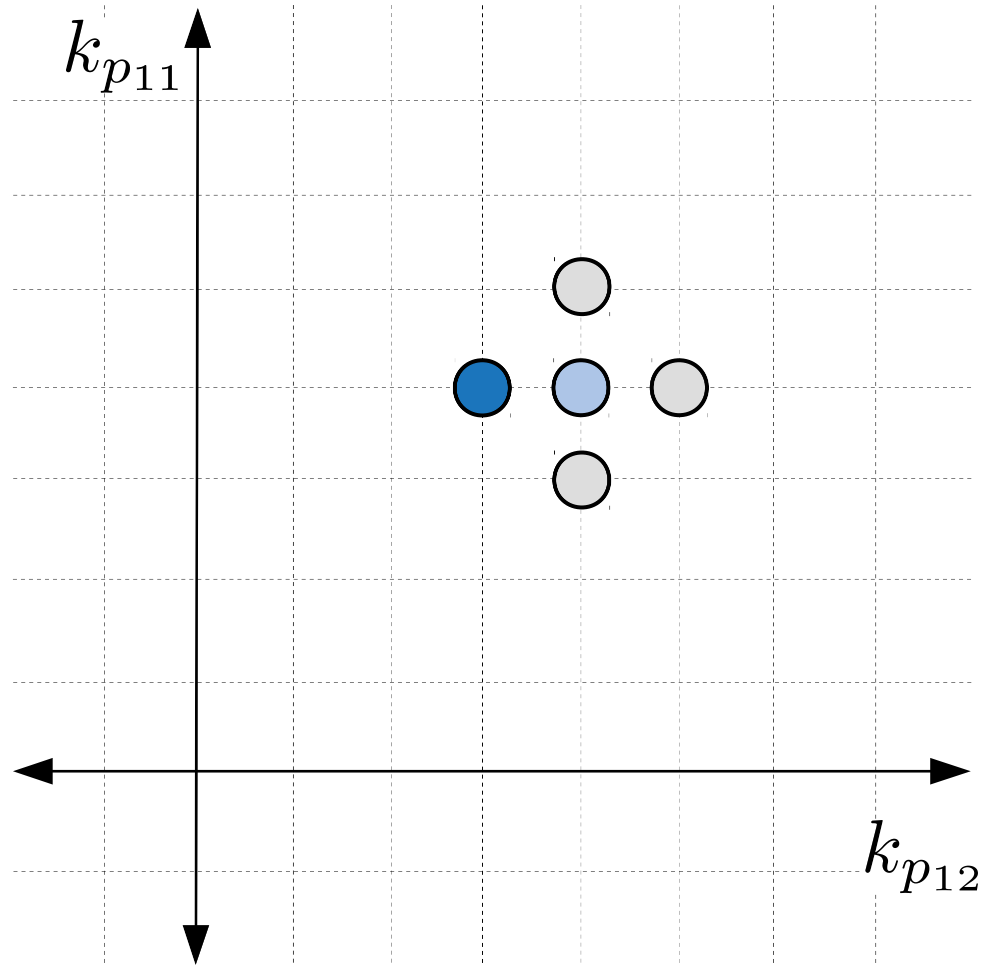
Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points
  (16 points in full 8D gain space)
- Find minimum performance measure.
- Move to new minimum, **then recompute** performance measure at all neighboring points.
- Repeat algorithm...



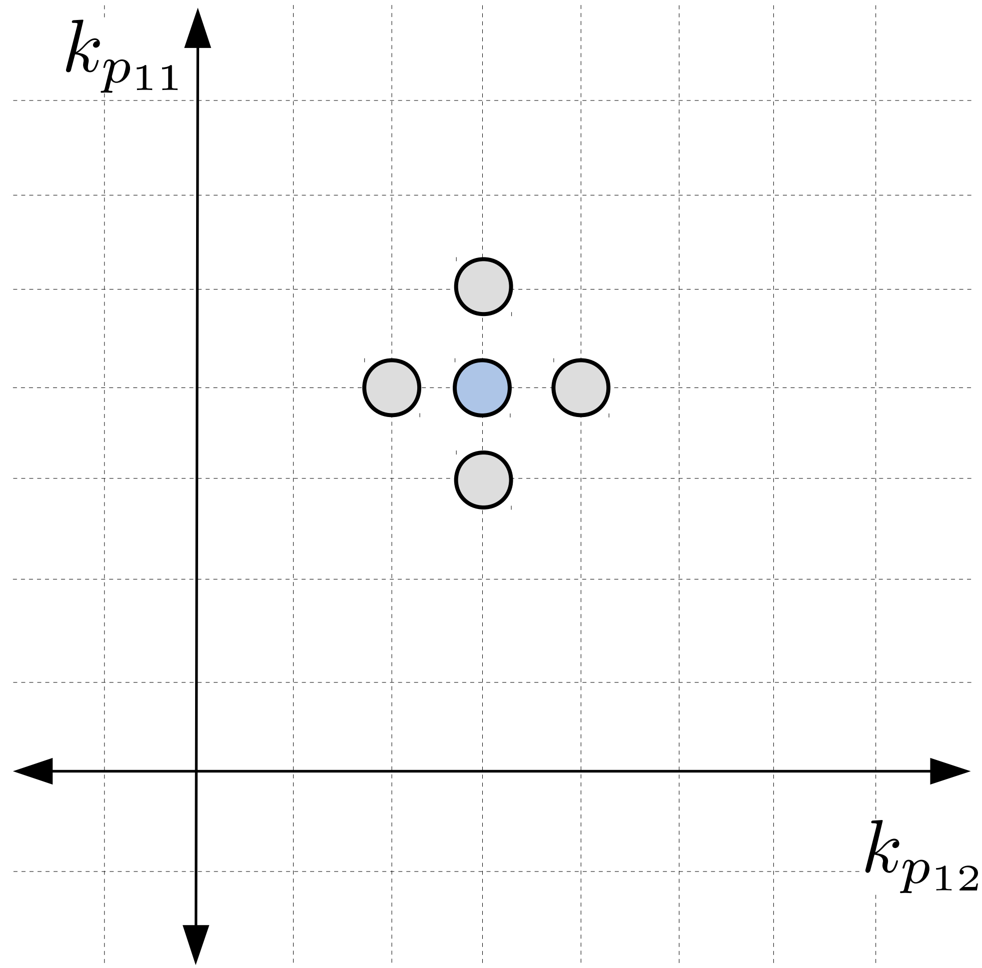
Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points
  (16 points in full 8D gain space)
- Find minimum performance measure.
- Move to new minimum, **then recompute** performance measure at all neighboring points.
- Repeat algorithm...



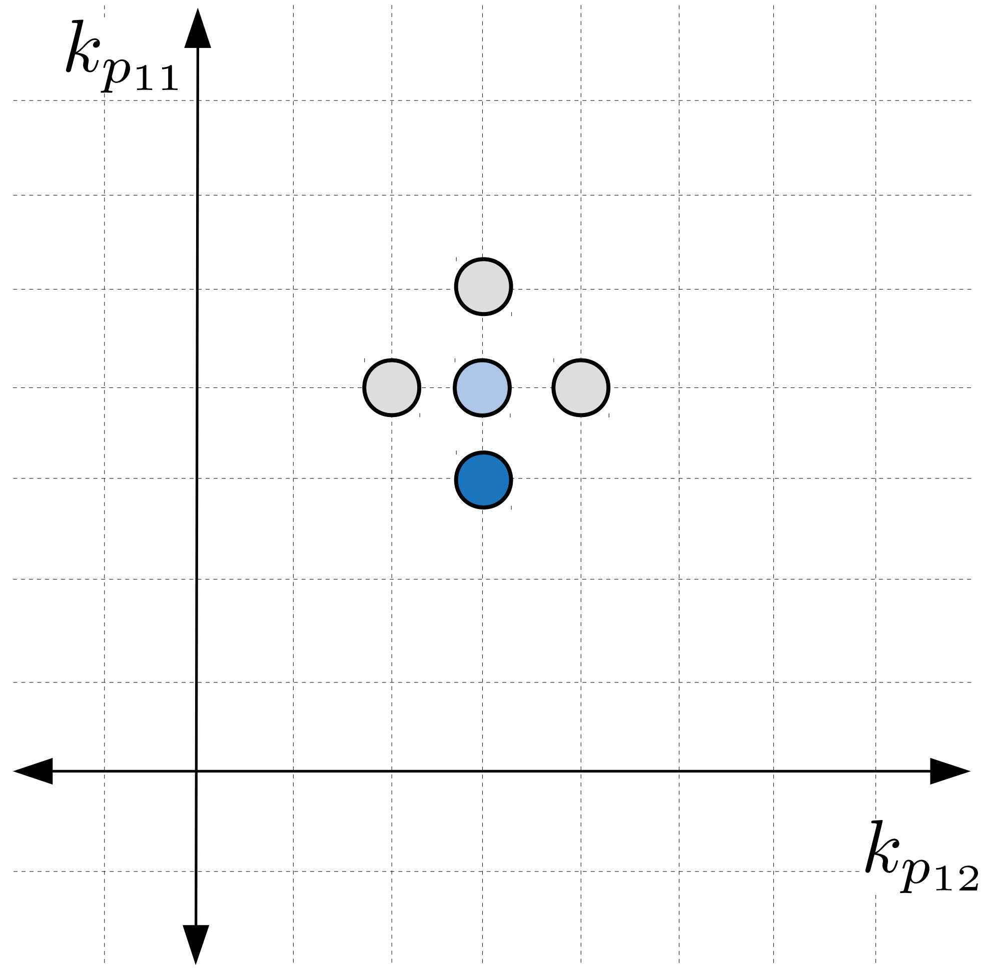
Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points
  (16 points in full 8D gain space)
- Find minimum performance measure.
- Move to new minimum, **then recompute** performance measure at all neighboring points.
- Repeat algorithm...



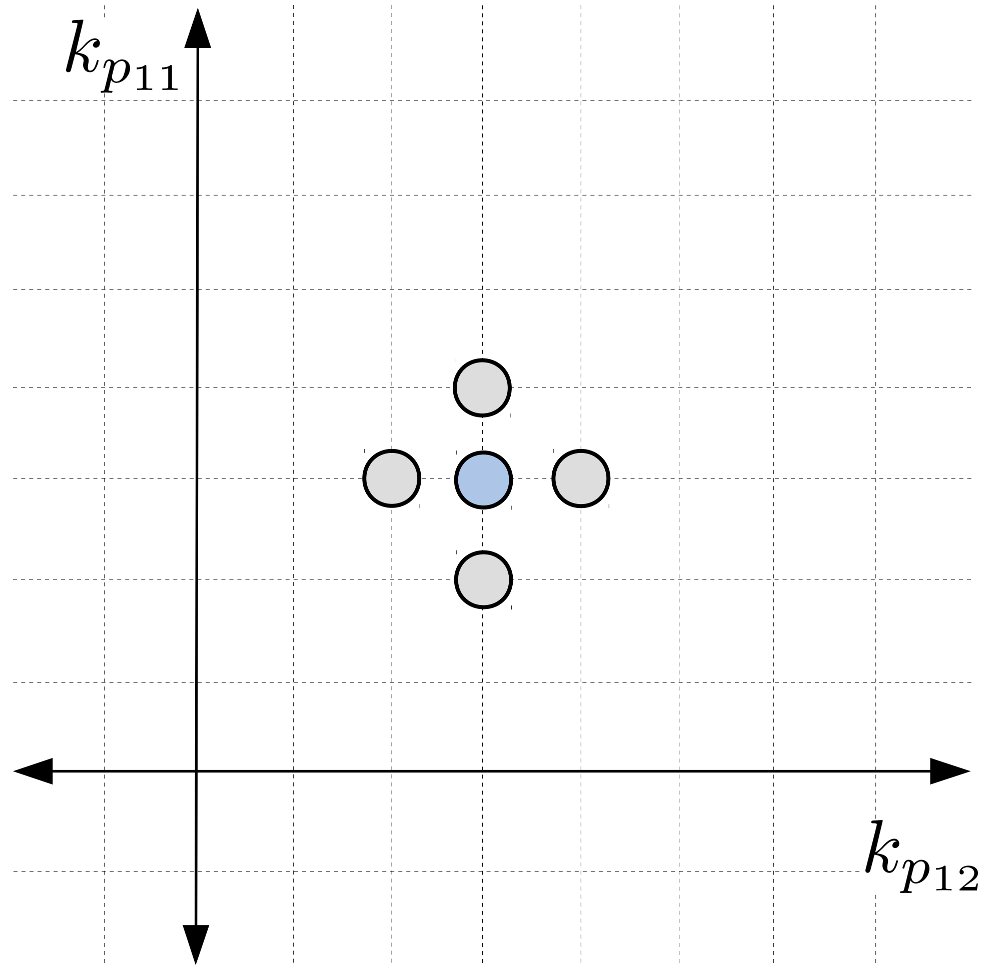
Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points
  (16 points in full 8D gain space)
- Find minimum performance measure.
- Move to new minimum, **then recompute** performance measure at all neighboring points.
- Repeat algorithm...



Basic Pattern Search Algorithm:

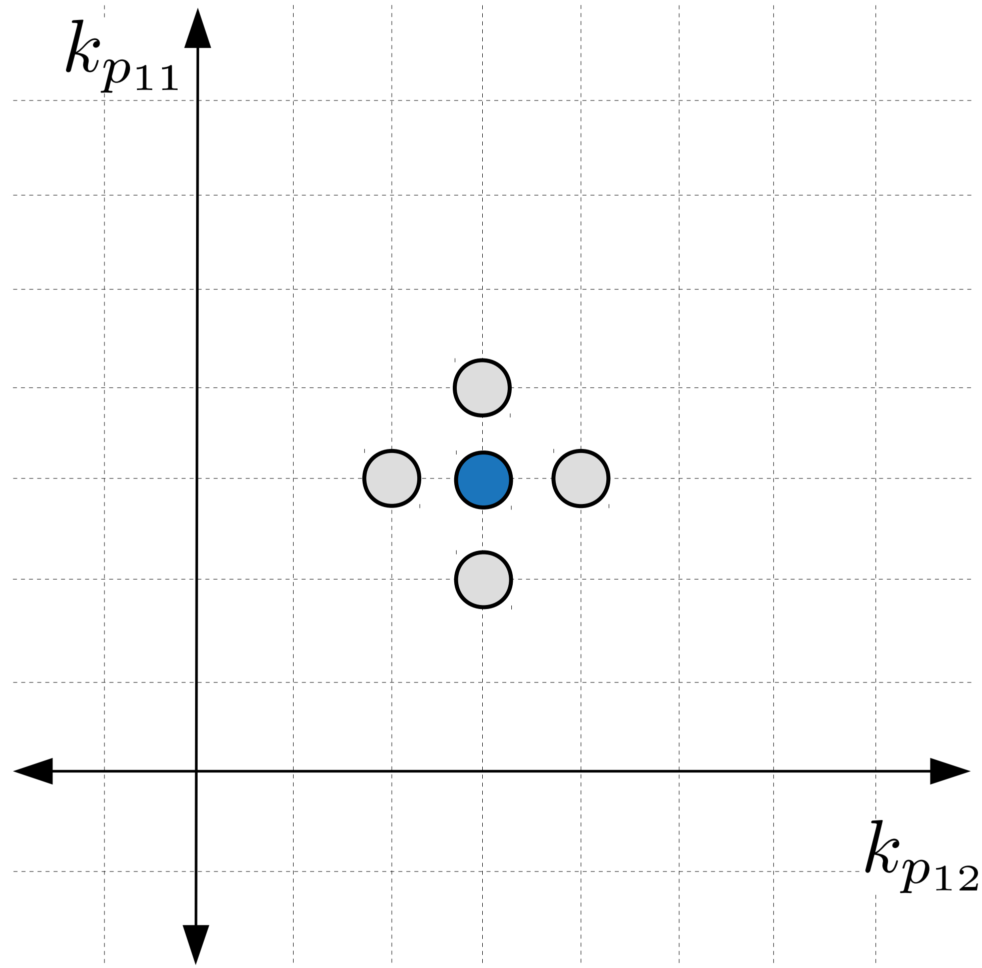
- Compute performance measure at starting point.
- Compute performance measure at all neighboring points
  (16 points in full 8D gain space)
- Find minimum performance measure.
- Move to new minimum, **then recompute** performance measure at all neighboring points.
- Repeat algorithm...



Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points

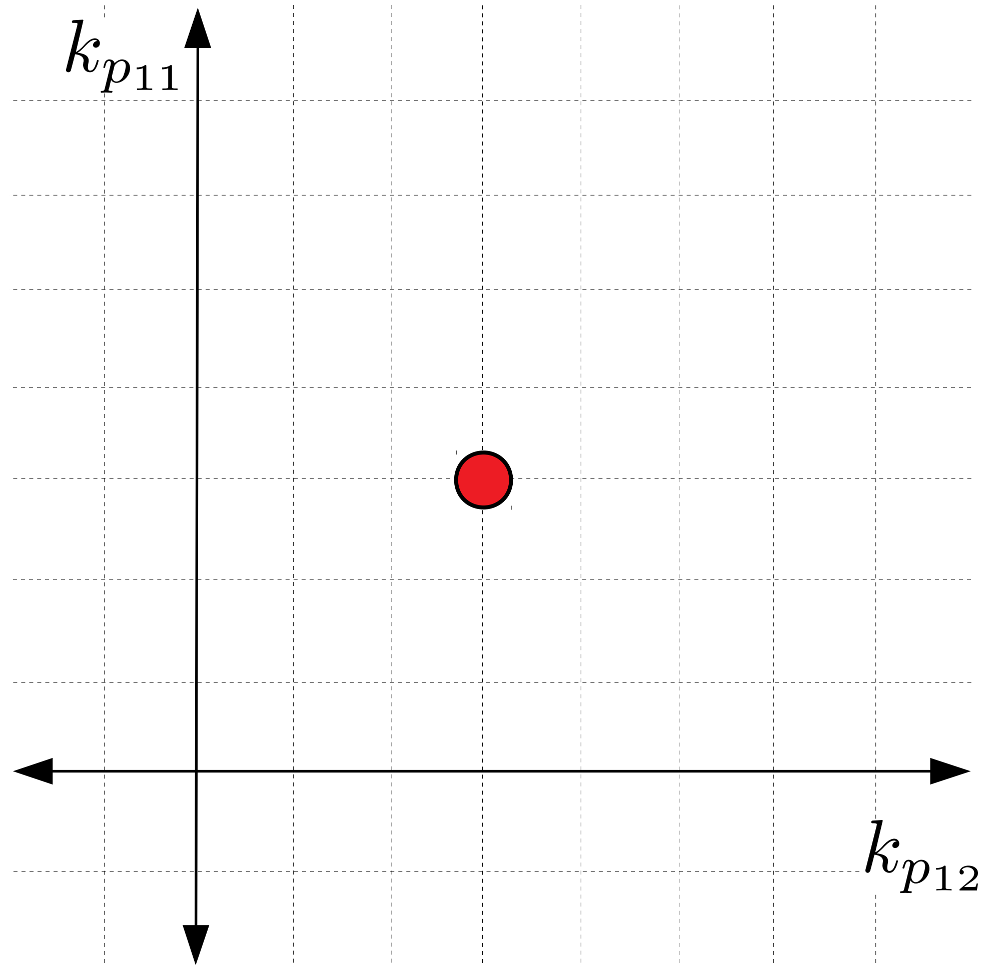
  (16 points in full 8D gain space)
- Find minimum performance measure.
- Move to new minimum, **then recompute** performance measure at all neighboring points.
- Repeat algorithm...



Basic Pattern Search Algorithm:

- Compute performance measure at starting point.
- Compute performance measure at all neighboring points

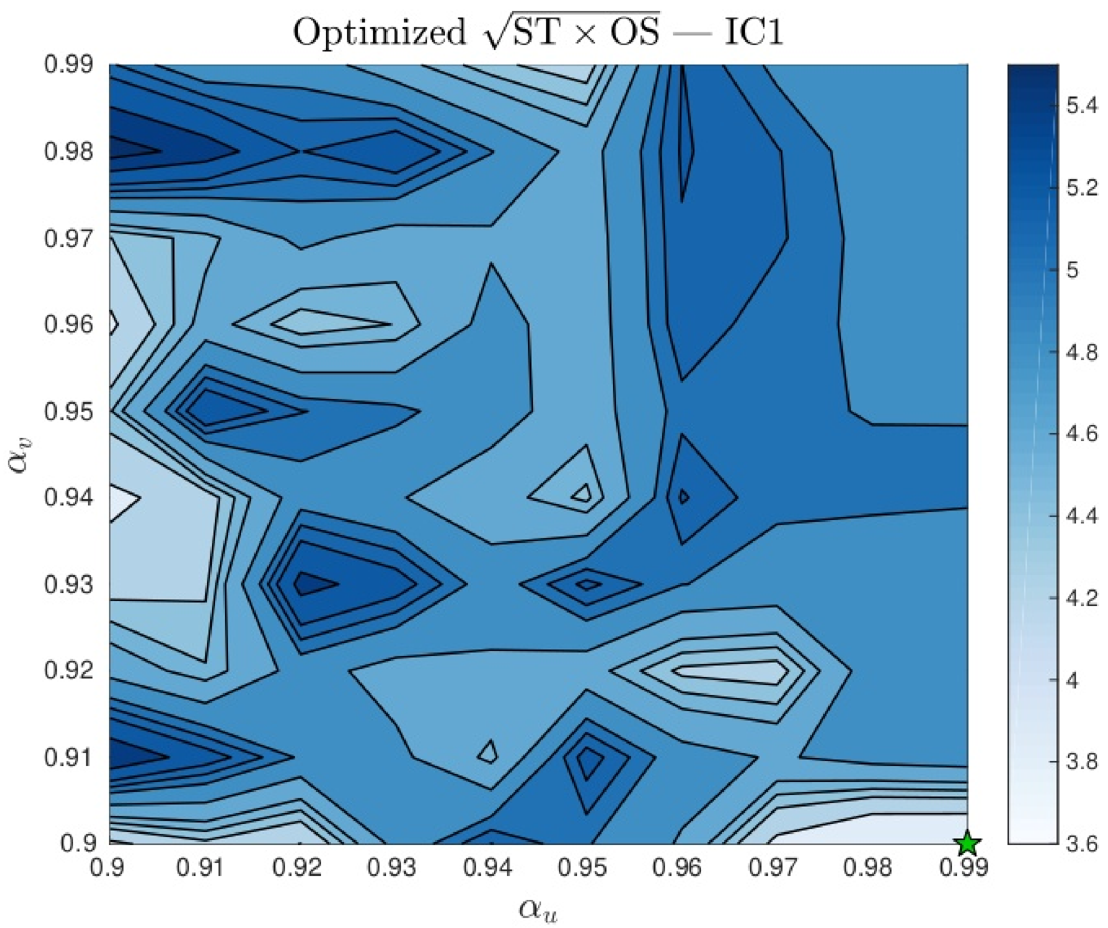
  (16 points in full 8D gain space)
- Find minimum performance measure.
- Move to new minimum, **then recompute** performance measure at all neighboring points.
- Repeat algorithm…



When neighboring points do not further minimize performance measure, **the optimal point in gain space is reached.**

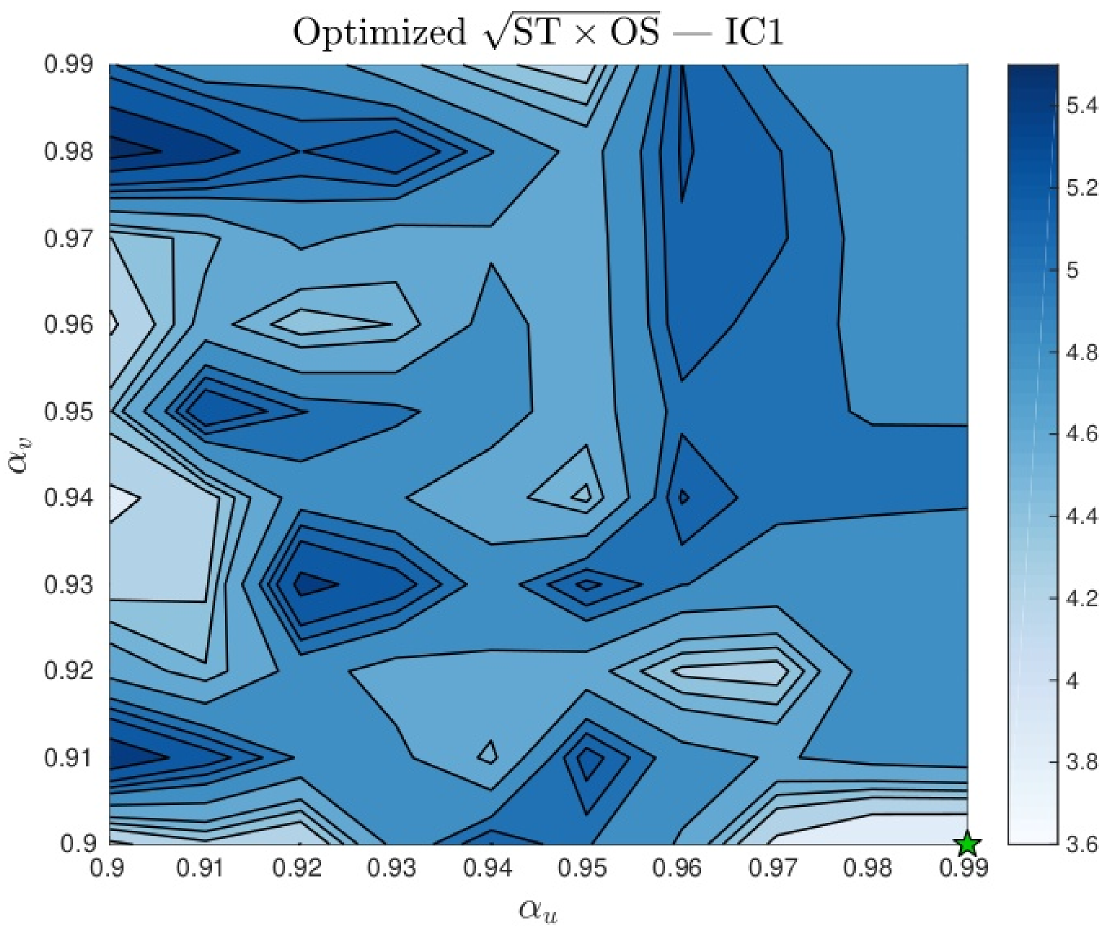
Do this for all grid points in 2D fractional-order space, $\{\alpha_u, \alpha_v\}$

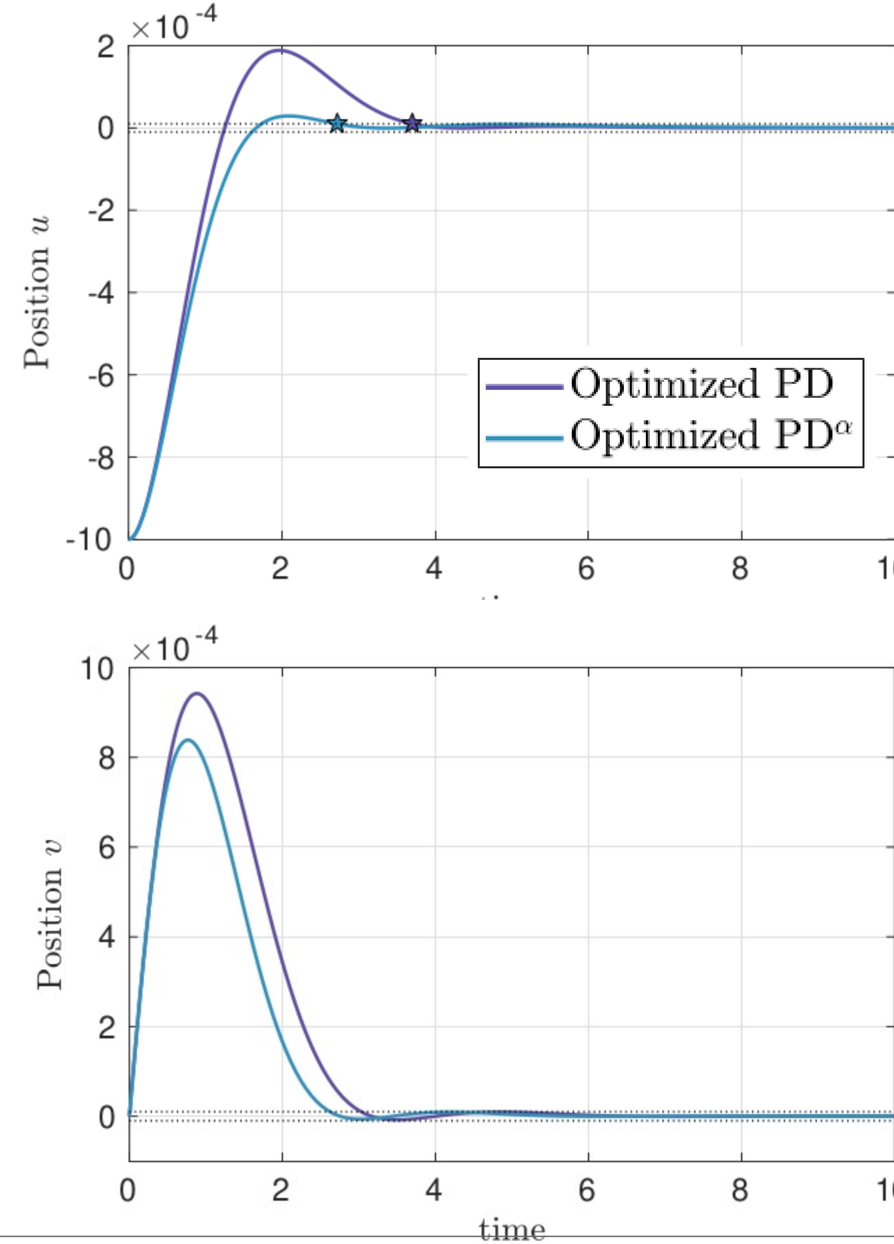
Simultaneously optimizing both ST and OS: $\psi = \sqrt{\mathrm{ST} \times \mathrm{OS}}$



Settling time reduced by 26%
Overshoot reduced by 20%

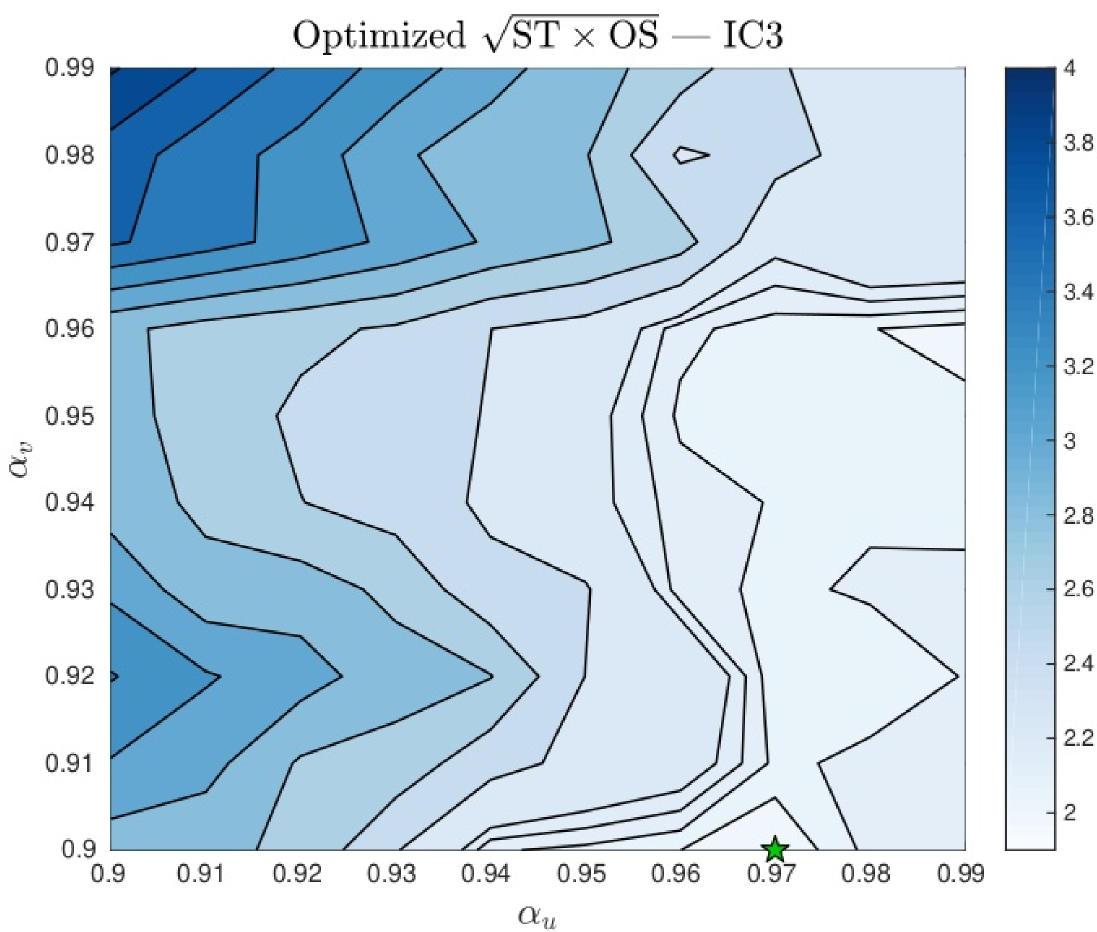
Simultaneously optimizing both ST and OS: $\psi = \sqrt{\mathrm{ST} \times \mathrm{OS}}$

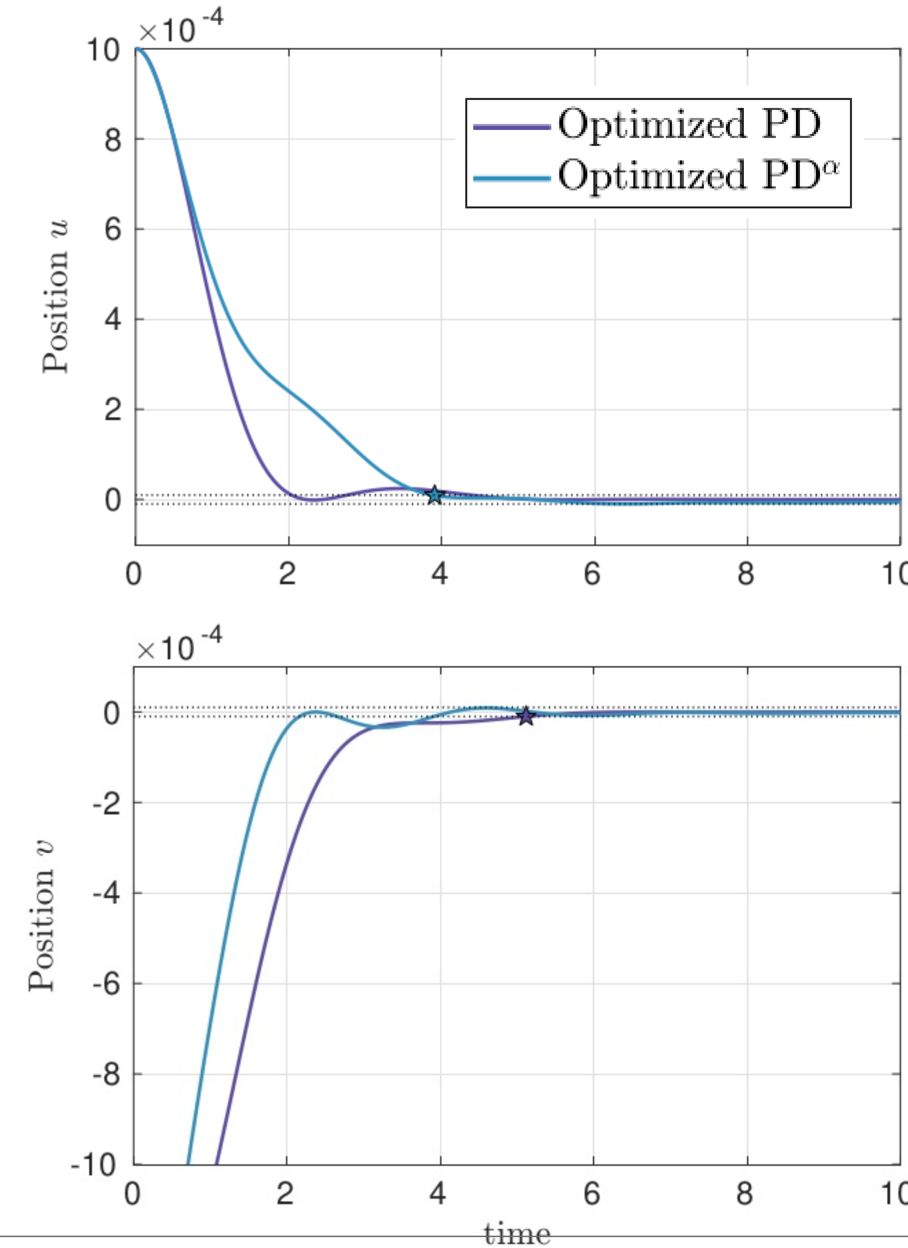




Settling time reduced by 26%
Overshoot reduced by 20%

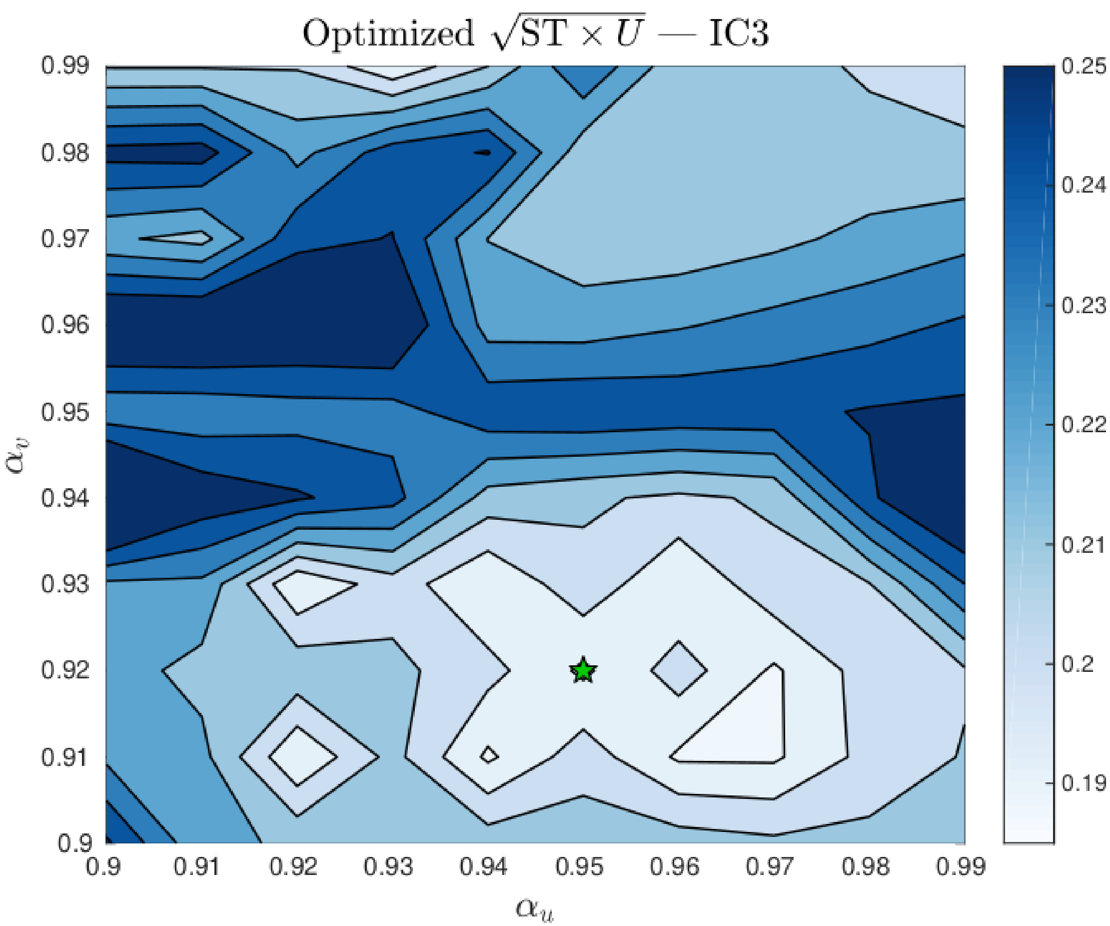
Simultaneously optimizing both ST and OS: $\psi = \sqrt{\mathrm{ST} \times \mathrm{OS}}$

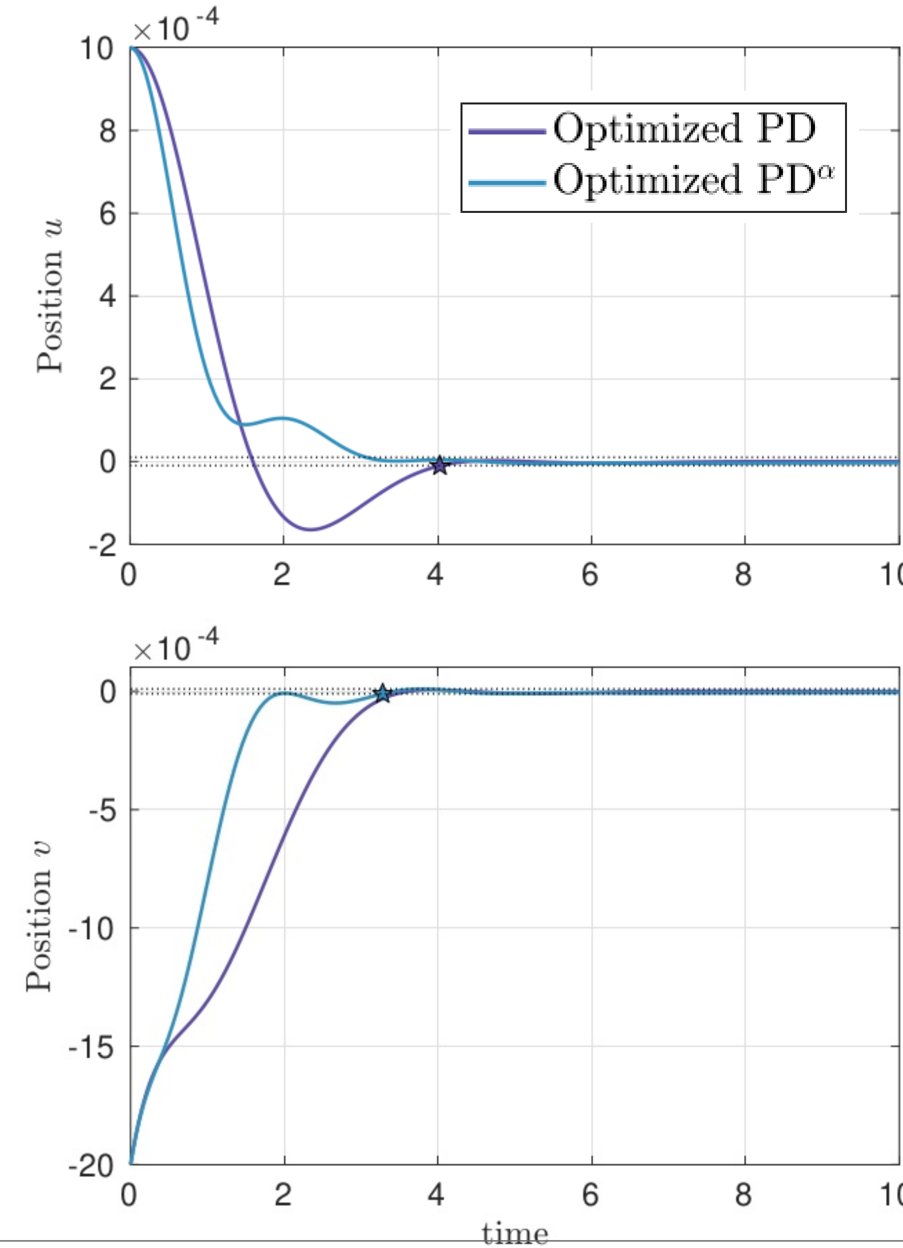




Settling time reduced by 24%
Overshoot *increased* by 1%

Simultaneously optimizing both ST and U: $\psi = \sqrt{\mathrm{ST} \times U}$





Settling time reduced by 18%
Control effort essentially unchanged

Conclusions

- We have generalized standard PD control for HCW dynamics by introducing fractional derivative terms

- Fractionally-controlled trajectory has additional degrees of freedom, *e.g.*,

$$\ddot{z} + b\dot{z} + az = 0$$ Standard controller: *a*, *b*

$$\ddot{z} + bD^{\alpha}z + az = 0$$ Fractional controller: *a*, *b*, *α*

- More **optimal trajectories** can be achieved by tuning this additional parameter.
  - Qualitative: More direct rendezvous (less variation in approach direction)
  - Quantitative: 27% reduction in settling time with less fuel cost.
  - More optimal behavior can be achieved in **both in-plane and out-of-plane motion**.