

PIPG for Rendezvous

Govind Chari

Univeristy of Washington

May 2023



PIPG

- Is a first order primal-dual conic optimizer
- Is easily verifiable
- Is factorization free
- Is an intuitive algorithm
- Is very fast (3-7x faster than ECOS)
- Can be coded in an afternoon
- Supports infeasibility detection

Comparison with ECOS

	ECOS	PIPG
Verification Ease	✗	✓
Factorization Free	✗	✓
Understandable	✗	✓
Implementation Ease	✗	✓
Infeasibility Detection	✓	✓
Tuning Free	✓	✗

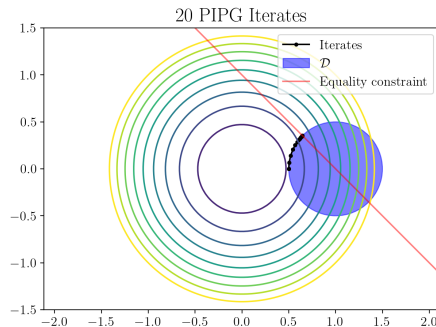
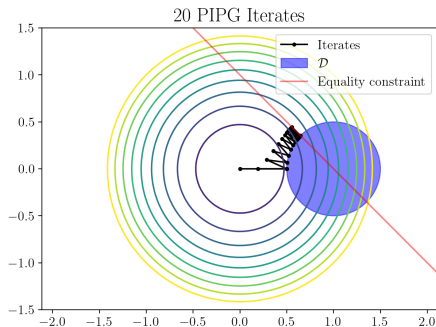
PIPG Algorithm

Optimization Problem

$$\begin{aligned} \min_z \quad & \frac{1}{2} z^\top P z + q^\top z \\ \text{s.t.} \quad & H z - h = 0 \\ & z \in \mathbb{D} \end{aligned}$$

Algorithm

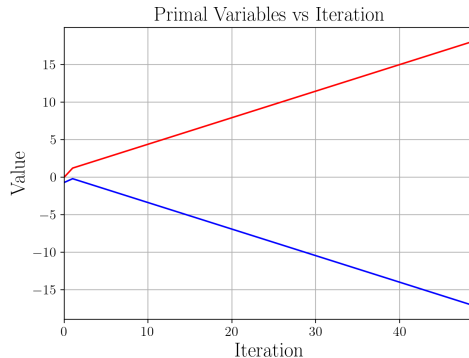
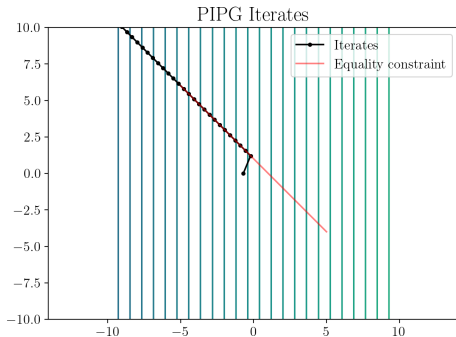
$$\begin{aligned} z^{k+1} &\leftarrow \Pi_{\mathbb{D}} \left[z^k - \alpha (P z^k + q + H^\top w^k) \right] \\ w^{k+1} &\leftarrow w^k + \beta (H(2z^{k+1} - z^k) - h) \end{aligned}$$



Infeasibility Detection - Dual Infeasibility

Dual Infeasibility Criteria:

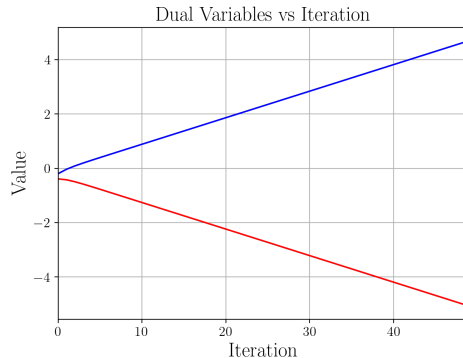
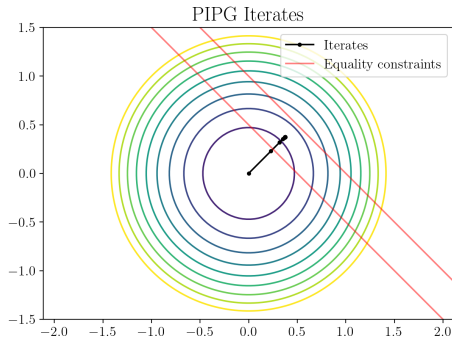
$$\lim_{k \rightarrow \infty} \|z^{k+1} - z^k\| \neq 0$$



Infeasibility Detection - Primal Infeasibility

Primal Infeasibility Criteria:

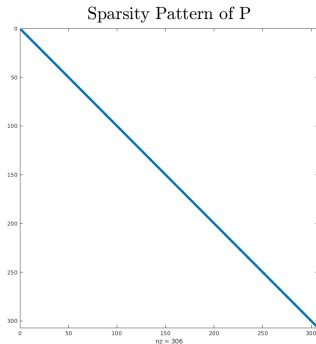
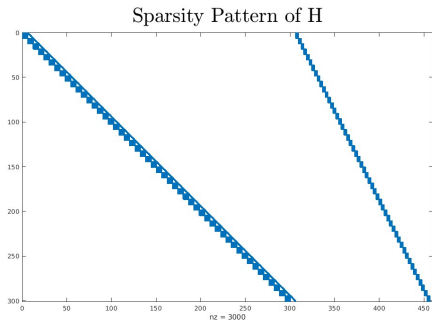
$$\lim_{k \rightarrow \infty} \|w^{k+1} - w^k\| \neq 0$$



Structured Sparsity

$$\begin{aligned} & \underset{x_t, u_t}{\text{minimize}} && x_N^\top Q_N x_N + \sum_{t=1}^{N-1} x_t^\top Q_t x_t + u_t^\top R_t u_t \\ & \text{subject to} && x_{t+1} = A_t x_t + B_t u_t, \\ & && x_t \in \mathbb{X}_t, \quad u_t \in \mathbb{U}_t \end{aligned}$$

$$\begin{aligned} & \underset{z}{\text{minimize}} && \frac{1}{2} z^\top P z \\ & \text{subject to} && H z - h = 0, \quad z \in \mathbb{D} \end{aligned}$$



Customization

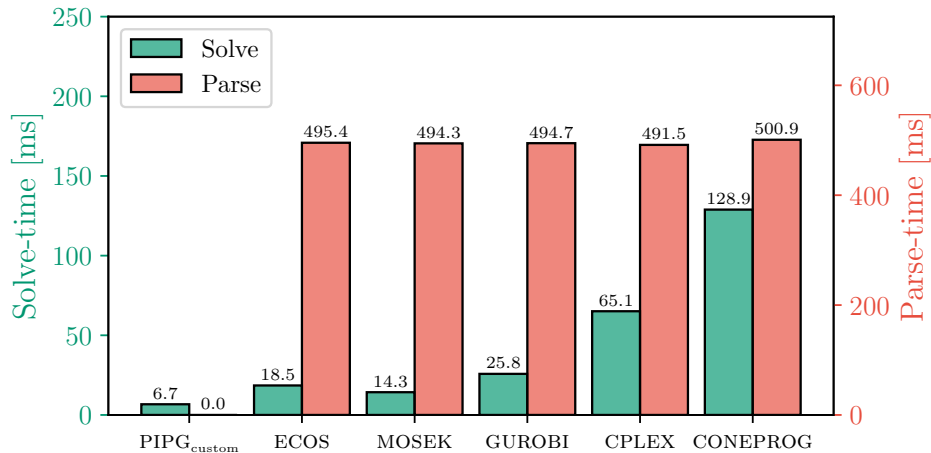
$$\begin{aligned} & \underset{x_t, u_t}{\text{minimize}} && x_N^\top Q_N x_N + \sum_{t=1}^{N-1} x_t^\top Q_t x_t + u_t^\top R_t u_t \\ & \text{subject to} && x_{t+1} = A_t x_t + B_t u_t, \\ & && x_t \in \mathbb{X}_t, \quad u_t \in \mathbb{U}_t \end{aligned}$$

$$\begin{aligned} & \underset{z}{\text{minimize}} && \frac{1}{2} z^\top P z \\ & \text{subject to} && H z - h = 0, \quad z \in \mathbb{D} \end{aligned}$$

$$\begin{aligned} w_t &\leftarrow v_t + \beta(x_{t+1} - A_t x_t - B_t u_t) \\ u_t &\leftarrow \Pi_{\mathbb{U}_t}[u_t - \alpha(R_t u_t - B_t^\top w_{t-1})] \\ x_t &\leftarrow \Pi_{\mathbb{X}_t}[x_t - \alpha(Q_t x_t + w_{t-1} - A_t^\top w_t)] \\ v_t &\leftarrow v_t + \beta(x_{t+1} - A_t x_t - B_t u_t) \end{aligned}$$

$$\begin{aligned} w &\leftarrow v + \beta(Hz - h) \\ z &\leftarrow \Pi_{\mathbb{D}}[z - \alpha(Pz + H^\top w)] \\ v &\leftarrow v + \beta(Hz - h) \end{aligned}$$

Parsing



From Elango et al. *SciTech 2021*

Continuous-time Nonconvex Problem

$$\underset{t_f, x(t), u(t)}{\text{minimize}} \quad \int_0^{t_f} \|u(t)\|_2^2 dt$$

$$\text{subject to} \quad \forall t \in [0, t_f)$$

CW Dynamics

$$\dot{x}(t) = f(x(t), u(t))$$

Max delta-v

$$\|u(t)\|_2 \leq u_{\max}$$

Keepout Zone

$$\|r(t) - r_c\|_2 \geq \rho$$

Max Speed

$$\|v(t)\|_2 \leq v_{\max}$$

Initial Conditions

$$r(0) = r_i$$

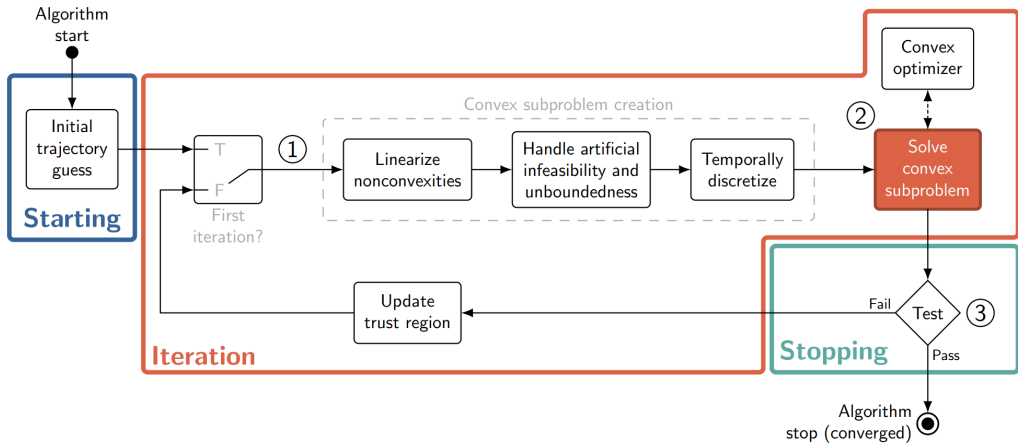
$$v(0) = v_i$$

Terminal Conditions

$$r(t_f) = 0_{3 \times 1}$$

$$v(t_f) = 0_{3 \times 1}$$

Successive Convex Programming (SCP)



Discrete-time Convex Subproblem

$$\underset{\sigma, x, u}{\text{minimize}} \quad \sum_{k=1}^{K-1} \|u_k\|_2^2 + w_{tr} \left(\sum_{k=1}^K \|x_k - \hat{x}_k\|_2^2 + \sum_{k=1}^{K-1} [\|u_k - \hat{u}_k\|_2^2 + (\sigma_k - \hat{\sigma}_k)^2] \right) + w_{vc} \|\nu^c\|_1 + w_{vb} \sum_{k=1}^K \nu_k^b$$

subject to $\forall k \in [1, K]$

Discrete Dynamics

$$x_{k+1} = A_k x_k + B_k u_k + S_k \sigma_k + c_k + \nu_k^c$$

Dilation Constraints

$$\sigma_{\min} \leq \sigma_k \leq \sigma_{\max}$$

Max delta-v

$$\|u_k\|_2 \leq u_{\max}$$

Keepout Zone

$$\|\hat{r}_k - r_c\|_2 + \left(\frac{\hat{r}_k - r_c}{\|\hat{r}_k - r_c\|_2} \right)^\top (r_k - \hat{r}_k) + \nu_k^b \geq \rho$$

$$\nu_k^b \geq 0$$

Max Speed

$$\|v_k\|_2 \leq v_{\max}$$

Initial Conditions

$$r(0) = r_i$$

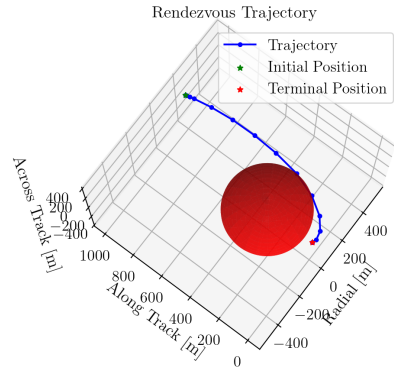
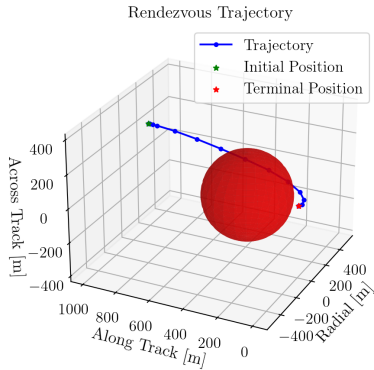
$$v(0) = v_i$$

Terminal Conditions

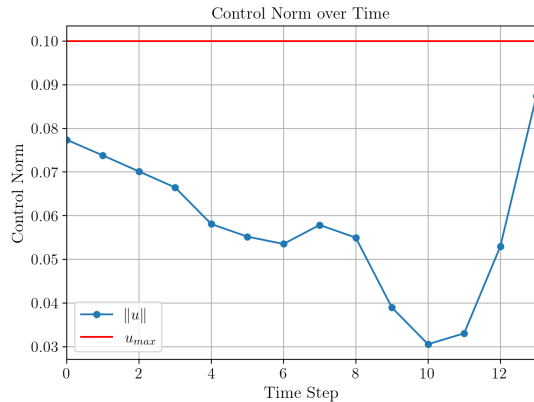
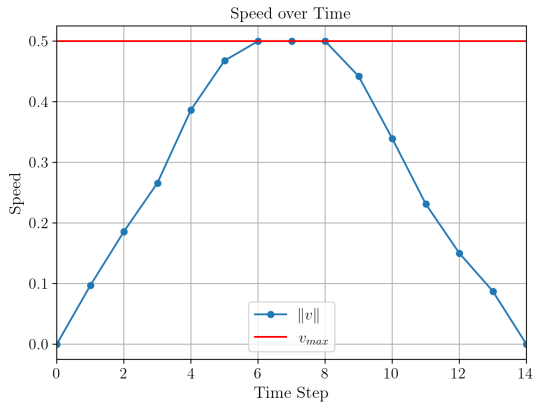
$$r(t_f) = 0_{3 \times 1}$$

$$v(t_f) = 0_{3 \times 1}$$

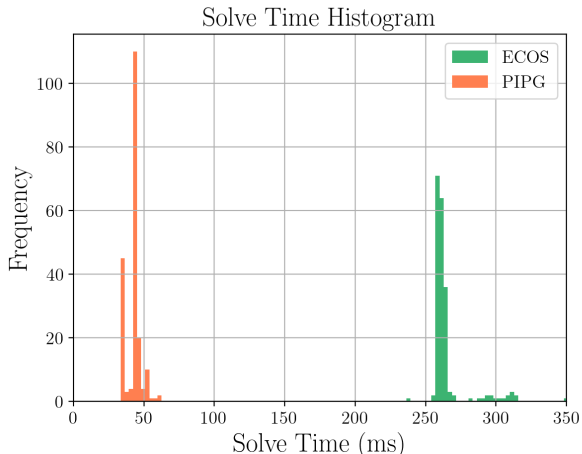
Trajectory



Trajectory



Speed Results



200 runs, 3 millisecond bins

	ECOS	PIPG
Mean	261ms	42ms
Median	267ms	44ms
STD	23ms	5ms

PIPG is **6.25x** faster even without customization

PIPG

- Is easy to understand
- Is easy to verify
- Is easy to implement
- Is faster than ECOS