Methodology Development to Radically Improve the Computational Performance of Physics-Based Mobility Solutions

Dr. Paramsothy Jayakumar, Senior Technical Expert, Analytics

Tank Automotive Research, Development and Engineering Center, U.S. Army

Overview

I. Background & Objectives
II. Nonsmooth Contact Dynamics
III. Quadratic Optimization with Conic Constraints
IV. Preconditioning
V. Numerical Results
VI. Conclusions

I. Background & Objectives

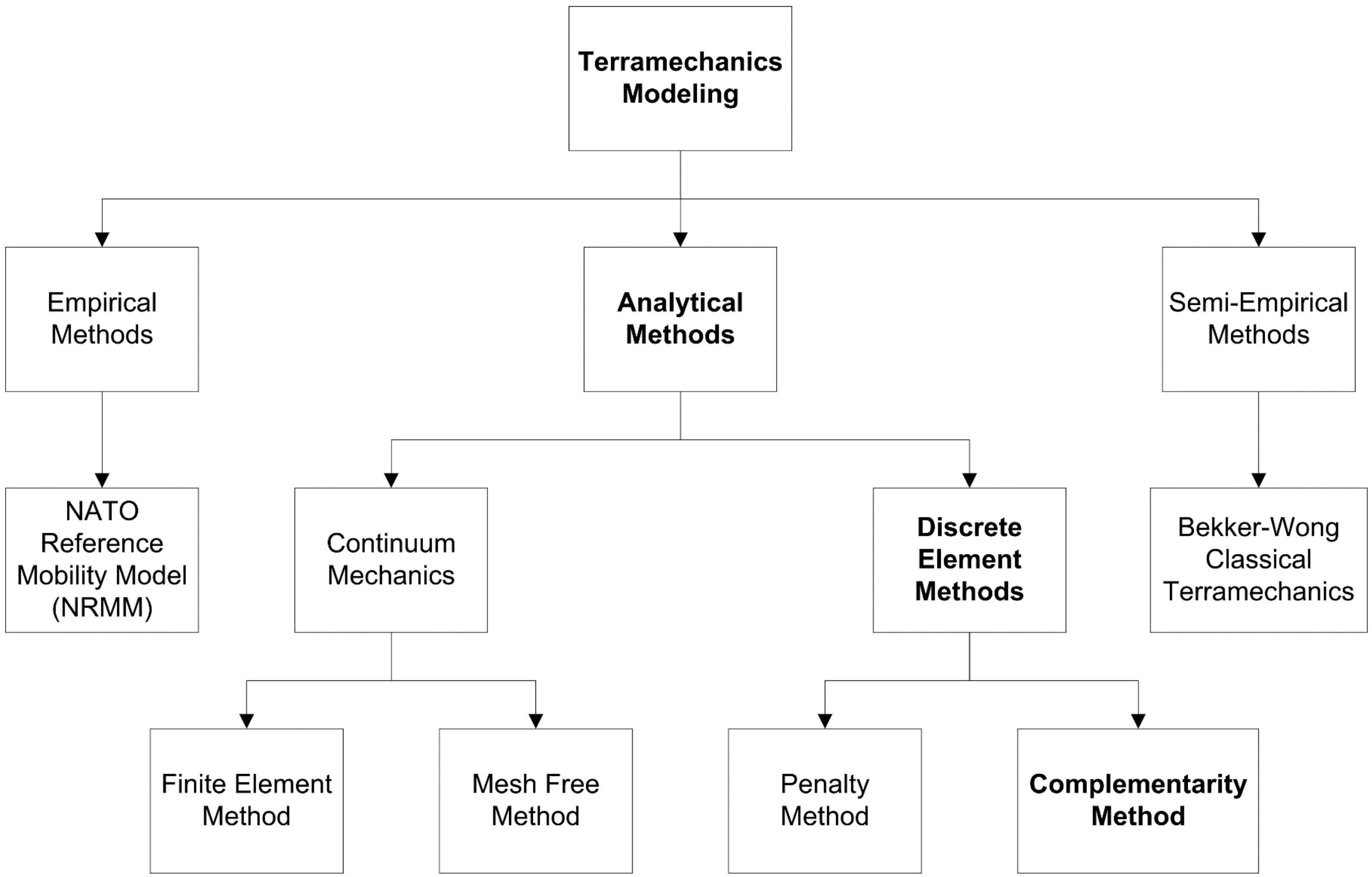
Background: Why terramechanics?

- Accurate and precise maneuvers provide the edge in combat for military units
- Scientists and engineers must design vehicles based on expected mobility challenges
- It is difficult and expensive to experimentally evaluate a vehicle's mobility performance
- Engineers and scientists are increasingly relying on modeling and simulation

Fig. 1: A vehicle maneuvering through mud.

Background: Current Methods

- There are several techniques that are used to study terramechanics



Examples of modeling techniques in terramechanics.

Background: Discrete Element Method

- The discrete element method (DEM) is a high fidelity terramechanics model capable of handling complex, heterogeneous soils with large deformations

Fig. 2: An immobilized military vehicle.

Fig. 3: DEM simulation of a HMMWV.

- DEM requires a large computational budget!

Objectives

- Augment the role that computational multibody dynamics plays in DEM applications. Several **barriers** are:
    i. The sheer **problem size**, which translates into hundreds of gigabytes of memory, teraflops of calculations, and terabytes of results
    ii. The decreasing **convergence speed** when handling increasingly large problems
    iii. The **multi-scale facet** that often emerges in discrete element applications such as vehicle mobility over deformable terrain

- To overcome these challenges, numerical methods that can leverage emerging commodity parallel computing hardware will be investigated
- **Higher than first-order methods** have the potential to **radically improve** the computational performance of the DEM method

II. Nonsmooth Contact Dynamics

The Complementarity Approach

- Two important concepts:

    1. Accounting for contact through complementarity

    2. Posing Coulomb Friction as an optimization problem

1) Accounting for contact through complementarity

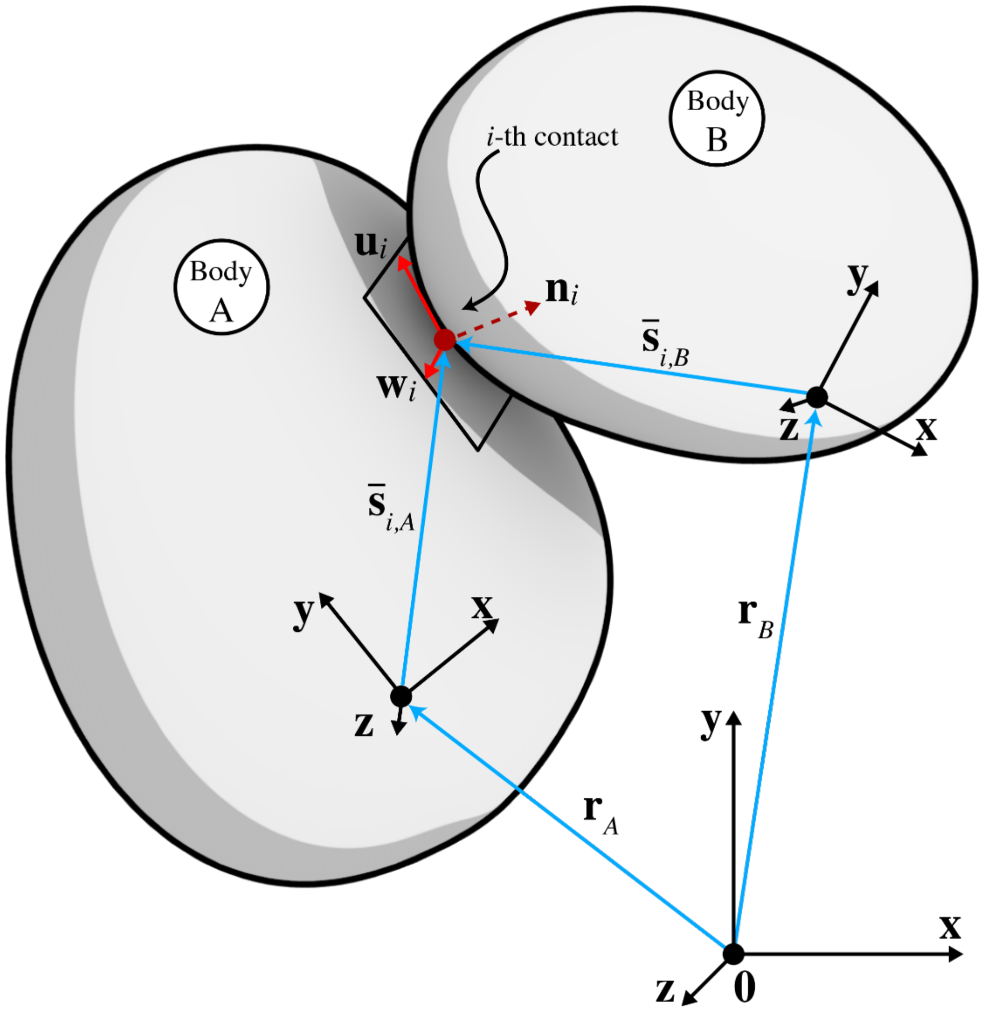


Fig. 4: Two bodies in contact.

- Two possible scenarios
  - The distance (gap) Φ between bodies is greater than zero, therefore the contact force γ_n is zero

or

  - The gap Φ between bodies is zero, therefore the contact force γ_n is non-zero

- One complementarity condition captures both scenarios:

$$0 \leq \Phi(\mathbf{q}_A, \mathbf{q}_B, t) \quad \perp \quad \gamma_{n,AB} \geq 0$$

2) Posing Coulomb friction as an optimization problem

- Actors in the Friction Force play, at a contact i:
  - Normal force γ_n
  - Friction coefficient μ
  - Relative slip velocity $\boldsymbol{v}_S$ at the contact point
  - Two orthogonal directions $\boldsymbol{d}_u$ and $\boldsymbol{d}_w$ spanning the contact tangent plane

- Coulomb friction model

$$\mu_i \widehat{\gamma}_{i,n} \geq \sqrt{\widehat{\gamma}_{i,u}^2 + \widehat{\gamma}_{i,w}^2}$$

$$\mathbf{F}_{i,T}^T \cdot \mathbf{v}_{i,T} = -\|\mathbf{F}_{i,T}\| \ \|\mathbf{v}_{i,T}\|$$

$$\|\mathbf{v}_{i,T}\| \left(\mu_i \widehat{\gamma}_{i,n} - \sqrt{\widehat{\gamma}_{i,u}^2 + \widehat{\gamma}_{i,w}^2}\right) = 0$$

- Components of friction force, γ_u and γ_w , found as solution of optimization problem

$$(\gamma_{i,u}, \gamma_{i,w}) = \underset{\sqrt{(\bar{\gamma}_u^i)^2 + (\bar{\gamma}_w^i)^2} \leq \mu_i \gamma_{i,n}}{\operatorname{argmin}} \mathbf{v}_{i,S}^T \cdot \left(\bar{\gamma}_u^i \mathbf{d}_{i,u} + \bar{\gamma}_w^i \mathbf{d}_{i,w}\right)$$

Complementarity Approach: The Math

$$
\begin{aligned}
\dot{\mathbf{q}} &= \mathbf{L}(\mathbf{q})\mathbf{v} \\
\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} &= \mathbf{f}(t,\mathbf{q},\mathbf{v}) - \mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q},t)\lambda + \sum_{i\in\mathcal{A}(\mathbf{q},\delta)} \underbrace{\left(\widehat{\gamma}_{i,n}\,\mathbf{D}_{i,n} + \widehat{\gamma}_{i,u}\,\mathbf{D}_{i,u} + \widehat{\gamma}_{i,w}\,\mathbf{D}_{i,w}\right)}_{i^{th}\text{frictional contact force}} \\
\mathbf{0} &= \mathbf{g}(\mathbf{q},t)
\end{aligned}
$$

$$
i \in \mathcal{A}(\mathbf{q}(t),\delta) \quad : \quad \left\{
\begin{array}{lll}
0 \le \Phi_i(\mathbf{q}) & \perp & \widehat{\gamma}_{i,n} \ge 0 \\
(\widehat{\gamma}_{i,u}, \widehat{\gamma}_{i,w}) & = & \underset{\sqrt{(\bar{\gamma}_u^i)^2 + (\bar{\gamma}_w^i)^2} \le \mu_i \widehat{\gamma}_{i,n}}{\operatorname{argmin}} \mathbf{v}^T \cdot \left(\bar{\gamma}_u^i\,\mathbf{D}_{i,u} + \bar{\gamma}_w^i\,\mathbf{D}_{i,w}\right)
\end{array}
\right.
$$

Time Stepping Scheme

$$\overbrace{\mathbf{q}^{(l+1)}}^{\text{Generalized positions}} = \mathbf{q}^{(l)} + \overbrace{h}^{\text{Step size}} \underbrace{\mathbf{L}}_{\text{Velocity transformation matrix}}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)}$$

$$\mathbf{M}(\overbrace{\mathbf{v}^{(l+1)}}^{\text{Generalized speeds}} - \mathbf{v}^{(l)}) = \underbrace{h\mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)})}_{\text{Applied impulse}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q}^{(l)}, t)\lambda}^{\text{Reaction impulse}} + \sum_{i \in \mathcal{A}(q^{(l)}, \delta)} \underbrace{(\gamma_{i,n}\mathbf{D}_{i,n} + \gamma_{i,u}\mathbf{D}_{i,u} + \gamma_{i,w}\mathbf{D}_{i,w})}_{\text{Frictional contact reaction impulses}}$$

$$0 = \underbrace{\frac{1}{h}\mathbf{g}(\mathbf{q}^{(l)}, t)}_{\text{Stabilization term}} + \mathbf{g_q}^T\mathbf{v}^{(l+1)} + \mathbf{g}_t$$

$$i \in \mathcal{A}(\mathbf{q}(t), \delta) \quad : \quad \begin{cases} 0 \le \overbrace{\frac{1}{h}\Phi_i(\mathbf{q}^{(l)})}^{\text{Stabilization term}} + \mathbf{D}_{i,n}^T\mathbf{v}^{(l+1)} \perp \gamma_{i,n} \ge 0 \\ (\gamma_{i,u}, \gamma_{i,w}) = \underset{\sqrt{(\bar{\gamma}_u^i)^2 + (\bar{\gamma}_w^i)^2} \le \mu_i \gamma_{i,n}}{\operatorname{argmin}} \mathbf{v}^{T,(l+1)} \cdot (\bar{\gamma}_u^i \mathbf{D}_{i,u} + \bar{\gamma}_w^i \mathbf{D}_{i,w}) \end{cases}$$

D.E. Stewart and J.C. Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *IJNME*, 39:2673-2691, 1996.

Nonlinear Complementarity to Cone Complementarity Problem

$$\overbrace{\mathbf{q}^{(l+1)}}^{\text{Generalized positions}} = \mathbf{q}^{(l)} + \overbrace{h}^{\text{Step size}} \underbrace{\mathbf{L}}_{\text{Velocity transformation matrix}}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)}$$

$$\mathbf{M}(\overbrace{\mathbf{v}^{(l+1)}}^{\text{Generalized speeds}} - \mathbf{v}^{(l)}) = \underbrace{h\mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)})}_{\text{Applied impulse}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q}^{(l)}, t)\lambda}^{\text{Reaction impulse}} + \sum_{i \in \mathcal{A}(q^{(l)}, \delta)} \underbrace{(\gamma_{i,n}\mathbf{D}_{i,n} + \gamma_{i,u}\mathbf{D}_{i,u} + \gamma_{i,w}\mathbf{D}_{i,w})}_{\text{Frictional contact reaction impulses}}$$

$$0 = \underbrace{\frac{1}{h}\mathbf{g}(\mathbf{q}^{(l)}, t)}_{\text{Stabilization term}} + {\mathbf{g}_{\mathbf{q}}}^{T}\mathbf{v}^{(l+1)} + \mathbf{g}_t$$

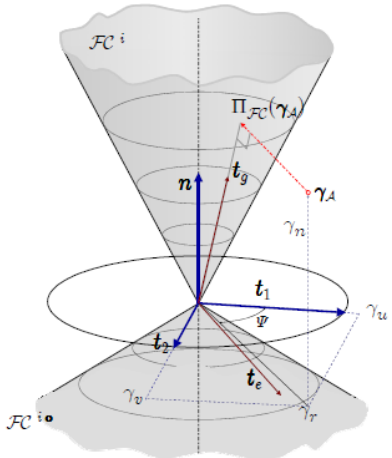
$$i \in \mathcal{A}(\mathbf{q}(t), \delta) \quad : \quad \begin{cases} 0 \le \overbrace{\frac{1}{h}\Phi_i(\mathbf{q}^{(l)})}^{\text{Stabilization term}} + \mathbf{D}_{i,n}^{T}\mathbf{v}^{(l+1)} - \overbrace{\mu_i \sqrt{\left(\boldsymbol{D}_{i,u}^{T}\mathbf{v}^{(l+1)}\right)^2 + \left(\boldsymbol{D}_{i,w}^{T}\mathbf{v}^{(l+1)}\right)^2}}^{Relaxation Term} \perp \gamma_{i,n} \ge 0 \\ (\gamma_{i,u}, \gamma_{i,w}) = \underset{\sqrt{(\bar{\gamma}_u^i)^2 + (\bar{\gamma}_w^i)^2} \le \mu_i \gamma_{i,n}}{\operatorname{argmin}} \quad \mathbf{v}^{T,(l+1)} \cdot (\bar{\gamma}_u^i \mathbf{D}_{i,u} + \bar{\gamma}_w^i \mathbf{D}_{i,w}) \end{cases}$$

M. Anitescu, Optimization-based Simulation of Nonsmooth Rigid Multibody Dynamics, Math. Program. 105 (1), 113-143, 2006

The Cone Complementarity Problem (CCP)

- Notation: for each contact i, $1 \le i \le N_c$, define the friction cone Υ_i and its polar cone Υ_i°:

$$\begin{aligned}\Upsilon_i &= \{[\gamma_{i,n}, \gamma_{i,u}, \gamma_{i,w}]^T \in \mathbb{R}^3 | \sqrt{\gamma_{i,u}^2 + \gamma_{i,w}^2} \le \mu_i \gamma_{i,n}\} \\ \Upsilon_i^\circ &= \{[\gamma_{i,n}, \gamma_{i,u}, \gamma_{i,w}]^T \in \mathbb{R}^3 | \gamma_{i,n} \le -\mu_i \sqrt{\gamma_{i,u}^2 + \gamma_{i,w}^2}\}\end{aligned}$$



- Notation: γ_i stores the contact $\gamma_{i,n}$ and friction $\gamma_{i,u}$ and $\gamma_{i,w}$ Lagrange multiplier

$$\gamma_i^T \equiv [\gamma_{i,n}, \gamma_{i,u}, \gamma_{i,w}]^T$$

- Notation: $\mathbf{N} \in \mathbb{R}^{3N_c \times 3N_c}$ and $\mathbf{r} \in \mathbb{R}^{3N_c}$ are two quantities computed at each time step of the simulation and which stay constant at that time step

- The Cone Complementarity Problem: find N_c Lagrange multipliers γ_i so that

$$\Upsilon_i \ni \gamma_i \perp -(\boldsymbol{N}\gamma + \mathbf{r})_i \in \Upsilon_i^\circ$$

The Optimization Angle

- CCP represents first order optimality condition of a quadratic problem with conic constraints

$$\begin{aligned} \mathbf{N} &= \mathbf{D}^T \mathbf{M}^{-1} \mathbf{D} \\ \mathbf{r} &= \mathbf{b} + \mathbf{D}^T \mathbf{M}^{-1} \boldsymbol{k} \\ \gamma &\equiv \left[\gamma_1^T, \gamma_2^T, \cdots, \gamma_{N_c}^T\right]^T \in \mathbb{R}^{3N_c} \end{aligned}$$

$$\gamma^\star = \underset{\substack{\gamma_i \in \Upsilon_i \\ 1 \le i \le N_c}}{\text{argmin}} \left(\frac{1}{2} \gamma^T \mathbf{N} \gamma + \mathbf{r}^T \gamma \right)$$

- $\mathbf{N} \in \mathbb{R}^{3N_c \times 3N_c}$ is symmetric and positive semi-definite
- $\mathbf{N}$ and $\mathbf{r} \in \mathbb{R}^{3N_c}$ do not depend on γ. They are computed once at beginning of each time step
- Problem has a global solution $\gamma^\star$
- Problem doesn't have a unique solution

Time Integration

- Life is simple once the frictional contact forces at the interface between shapes are available

  - Velocity at new time step *l+1* computed as

$$\mathbf{v}^{(l+1)} = \mathbf{M}^{-1}\left(\mathbf{k} + \mathbf{D}\gamma\right)$$

  - Once velocity available, the new set of generalized coordinates computed as

$$\mathbf{q}^{(l+1)} = \mathbf{q}^{(l)} + h\mathbf{L}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)}$$

Summary of the Complementarity Approach

- Complementarity conditions employed to link distance between particles and normal force
- Friction posed as an optimization problem
- Equations of motion became equilibrium constraints to the optimization problem
- DVI discretization leads to nonlinear complementarity problem
- Relaxation yields CCP, which was solved via a QP with conic constraints to compute γ

III. Quadratic Optimization with Conic Constraints

First-Order Methods

- Jacobi & Gauss-Seidel

ALGORITHM JACOBI($\boldsymbol{N}$, $\boldsymbol{r}$, τ, N_{max}, $\boldsymbol{\gamma}_0$)

(1) **for** $k := 0$ **to** N_{max}

(2) $\hat{\boldsymbol{\gamma}}_{(k+1)} = \Pi_{\mathcal{K}} \left(\boldsymbol{\gamma}_{(k)} - \omega \boldsymbol{B} \left(\boldsymbol{N} \boldsymbol{\gamma}_{(k)} + \boldsymbol{r} \right) \right)$

(3) $\boldsymbol{\gamma}_{(k+1)} = \lambda \hat{\boldsymbol{\gamma}}_{(k+1)} + (1 - \lambda) \boldsymbol{\gamma}_{(k)}$

(4) $r = r \left(\boldsymbol{\gamma}_{(k+1)} \right)$

(5) **if** $r < \tau$

(6) **break**

(7) **endfor**

(8) **return** Value at time step $t^{(l+1)}$, $\boldsymbol{\gamma}^{(l+1)} := \boldsymbol{\gamma}_{(k+1)}$

ALGORITHM GAUSS-SEIDEL($\boldsymbol{N}$, $\boldsymbol{r}$, τ, N_{max}, $\boldsymbol{\gamma}_0$)

(1) **for** $k := 0$ **to** N_{max}

(2) **for** $i = 1$ **to** n_c

(3) $\hat{\boldsymbol{\gamma}}_{i,(k+1)} = \Pi_{\mathcal{K}} \left(\boldsymbol{\gamma}_{i,(k)} - \omega \boldsymbol{B}_i \left(\boldsymbol{N} \boldsymbol{\gamma}_k + \boldsymbol{r} \right)_i \right)$

(4) $\boldsymbol{\gamma}_{i,(k+1)} = \lambda \hat{\boldsymbol{\gamma}}_{i,(k+1)} + (1 - \lambda) \boldsymbol{\gamma}_{i,(k)}$

(5) **endfor**

(6) $r = r \left(\boldsymbol{\gamma}_{k+1} \right)$

(7) **if** $r < \tau$

(8) **break**

(9) **endfor**

(10) **return** Value at time step $t^{(l+1)}$, $\boldsymbol{\gamma}^{(l+1)} := \boldsymbol{\gamma}_{(k+1)}$

- APGD

ALGORITHM NAPG($\boldsymbol{N}, \boldsymbol{r}, t \leq \frac{1}{\lambda_{max}(\boldsymbol{N})}, \tau, N_{max}$)

$\boldsymbol{\gamma}_0 = \boldsymbol{0}_{n_c}$

$\hat{\boldsymbol{\gamma}}_0 = \boldsymbol{1}_{n_c}$

$\boldsymbol{y}_0 = \boldsymbol{\gamma}_0$

$\theta_0 = 1$

for $k := 0$ **to** N_{max}

  $\boldsymbol{g} = \boldsymbol{N}\boldsymbol{y}_k - \boldsymbol{r}$

  $\boldsymbol{\gamma}_{k+1} = \Pi_{\mathcal{K}}\left(\boldsymbol{y}_k - t\boldsymbol{g}\right)$

  $\theta_{k+1} = \frac{-\theta_k^2 + \theta_k\sqrt{\theta_k^2+4}}{2}$

  $\beta_{k+1} = \theta_k \frac{1-\theta_k}{\theta_k^2 + \theta_{k+1}}$

  $\boldsymbol{y}_{k+1} = \boldsymbol{\gamma}_{k+1} + \beta_{k+1}\left(\boldsymbol{\gamma}_{k+1} - \boldsymbol{\gamma}_k\right)$

  $\epsilon = \epsilon\left(\boldsymbol{\gamma}_{k+1}\right)$

  if $\epsilon < \tau$

    break

  endif

endfor

return Value at time step t_{l+1}, $\boldsymbol{\gamma}^{l+1} := \hat{\boldsymbol{\gamma}}$.

First-Order Methods

Fig. 5: A field with hills.

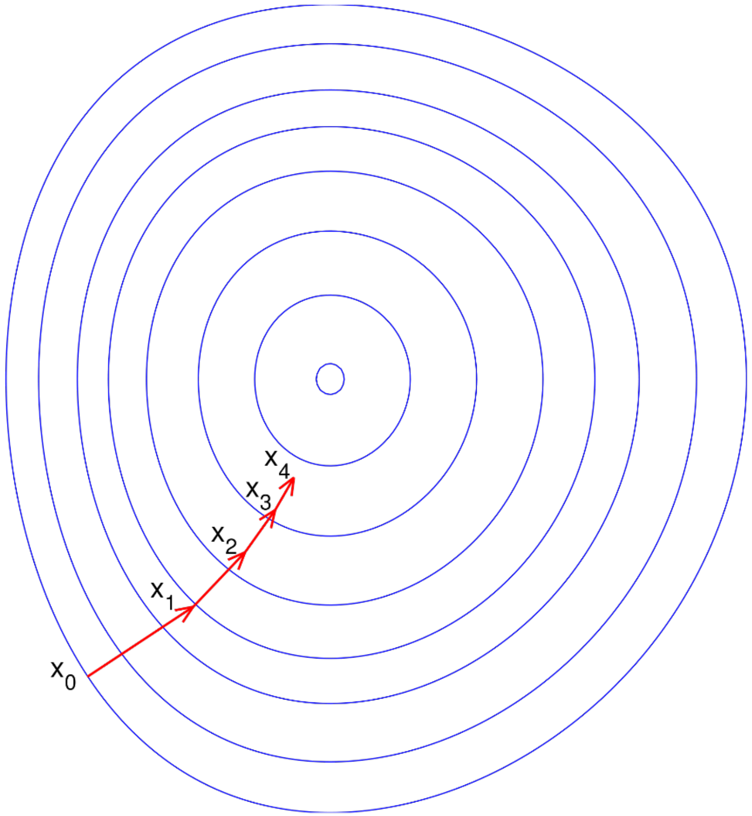


Fig. 6: The gradient descent method in 2D.

Second-Order Methods

- Original problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \le 0, \quad i = 1, \dots, m \\ & Ax = b \end{array}$$

- Reformulation via an indicator function:

$$\begin{array}{ll} \text{minimize} & f_0(x) + \sum_{i=1}^{m} I_-(f_i(x)) \\ \text{subject to} & Ax = b \end{array}$$

where $I_-(u) = 0$ if $u \le 0$, $I_-(u) = \infty$ otherwise

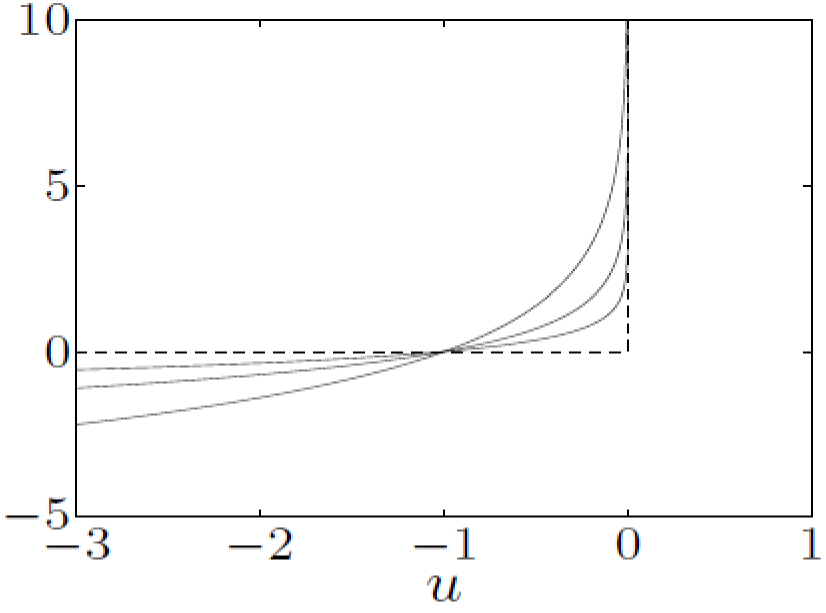


Fig. 7: The log-barrier method.

- Approximation via logarithmic barrier:

$$\begin{array}{ll} \text{minimize} & f_0(x) - (1/t) \sum_{i=1}^{m} \log(-f_i(x)) \\ \text{subject to} & Ax = b \end{array}$$

Interior Point Method

TARDEC

ALGORITHM PD-IP(f_0, f_1, ..., f_m, $\mu \geq 1$, ϵ)

(1) **while** $||\boldsymbol{r}_t(\boldsymbol{x}, \boldsymbol{\lambda})||_2 > \epsilon$

(2) Compute $t = \frac{\mu m}{\hat{\eta}}$

(3) Compute search direction $[\Delta \boldsymbol{x}^T \ \Delta \boldsymbol{\lambda}^T]^T$

(4) Compute step length $s > 0$ via line search

(5) Update: $\boldsymbol{x} = \boldsymbol{x} + s\Delta\boldsymbol{x}$, $\boldsymbol{\lambda} = \boldsymbol{\lambda} + s\Delta\boldsymbol{\lambda}$

(6) **endwhile**

(7) **return** Solution $\boldsymbol{x}^\star = \boldsymbol{x}$, $\boldsymbol{\lambda}^\star = \boldsymbol{\lambda}$

Hessian

$$\begin{bmatrix} \nabla^2 f_0(\boldsymbol{x}) + \sum_{i=1}^{m} \boldsymbol{\lambda}_i \nabla^2 \boldsymbol{f}_i(\boldsymbol{x}) & \nabla \boldsymbol{f}(\boldsymbol{x})^T \\ -diag(\boldsymbol{\lambda}) \nabla \boldsymbol{f}(\boldsymbol{x}) & -diag(\boldsymbol{f}(\boldsymbol{x})) \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{x} \\ \Delta \boldsymbol{\lambda} \end{bmatrix} = - \begin{bmatrix} \nabla f_0(\boldsymbol{x}) + \nabla \boldsymbol{f}(\boldsymbol{x})^T \boldsymbol{\lambda} \\ -diag(\boldsymbol{\lambda}) \boldsymbol{f}(\boldsymbol{x}) - \frac{1}{t}\mathbf{1} \end{bmatrix}$$

IV. Preconditioning with SaP::GPU

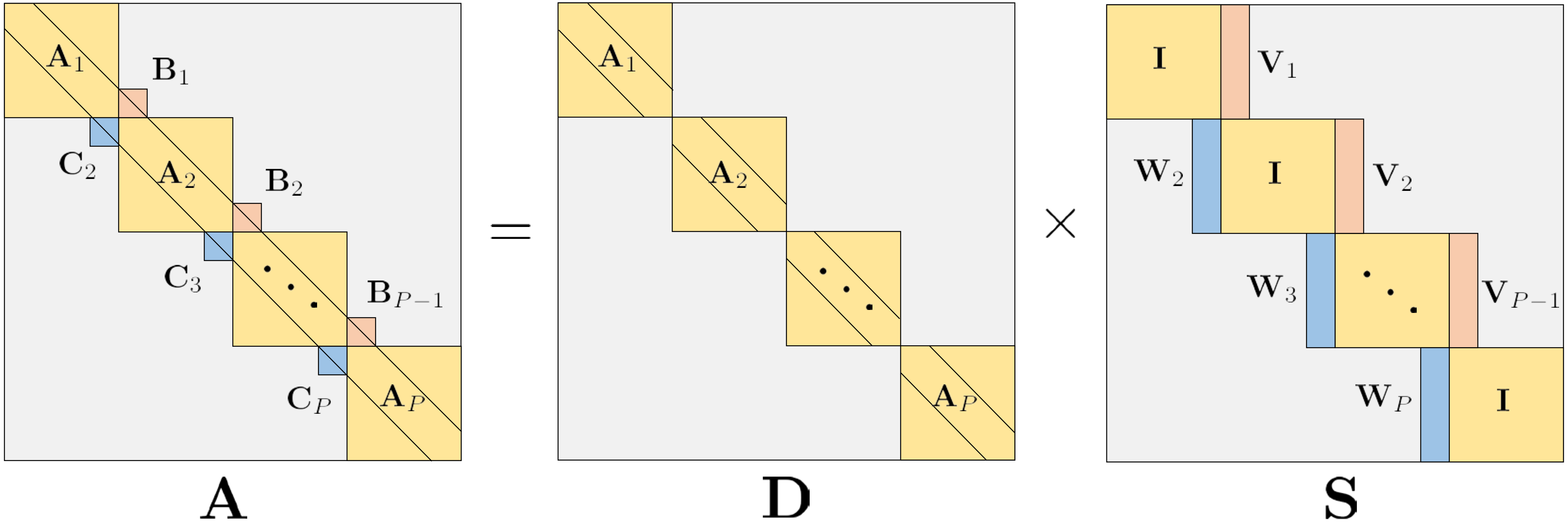
SaP::GPU: Background

- SPIKE: a divide-and-conquer approach to solving **banded** dense systems.
- Proposed by A. H. Sameh and D. J. Kuck in 1978.
  (see also E. Polizzi and A. H. Sameh, *Parallel Computing 32(2),* 2006)
- "Analysis of a Splitting Approach for the Parallel Solution of Linear Systems on GPU Cards," A. Li, R. Serban, D. Negrut
- Basic idea:
  - Partition the matrix $\mathbf{A}$.
  - Factorize $\mathbf{A}$ to isolate independent blocks.
  - Solve a reduced system to account for coupling information.
  - Recover solution of original system.
- SPIKE comes in two main flavors:
  - **Full-SPIKE**: recursively solve an exact reduced system (direct solver for banded matrices).
  - **Truncated-SPIKE**: solve an approximate reduced system in one step (needs iterative refinement).

SaP::GPU: Algorithmic Details

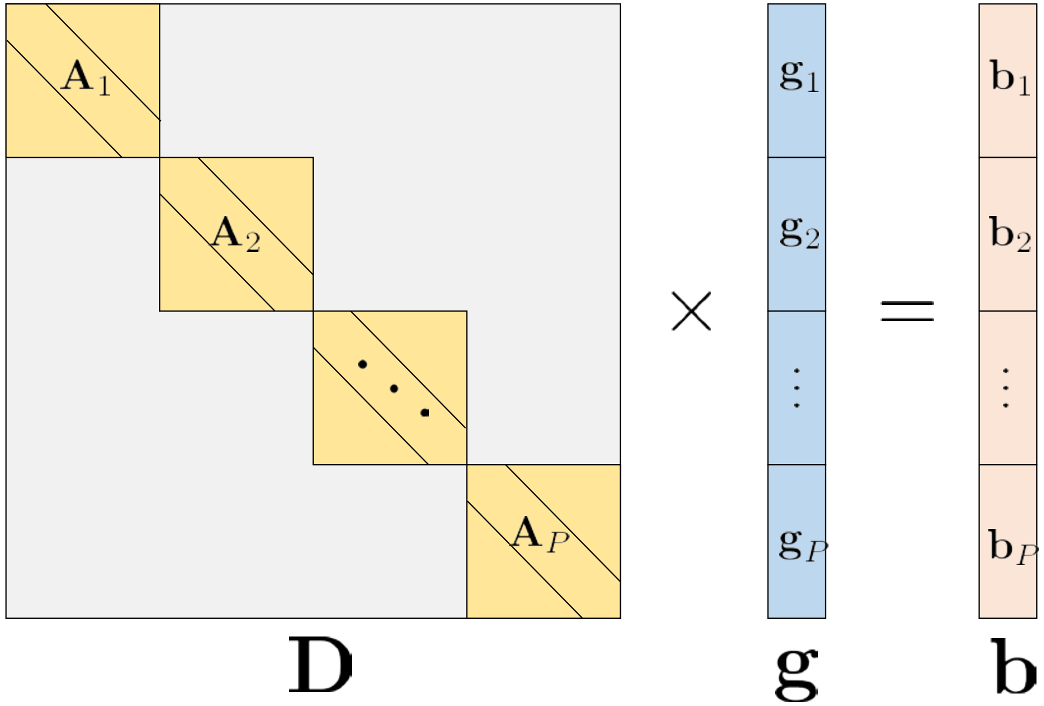
- Partitioning and Factorization
- Partition and factorize $\mathbf{A}$ into *block diagonal* matrix $\mathbf{D}$ and *spike* matrix $\mathbf{S}$.

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \Leftrightarrow \quad \begin{cases} \mathbf{D}\mathbf{g} = \mathbf{b} \\ \mathbf{S}\mathbf{x} = \mathbf{g} \end{cases}$$



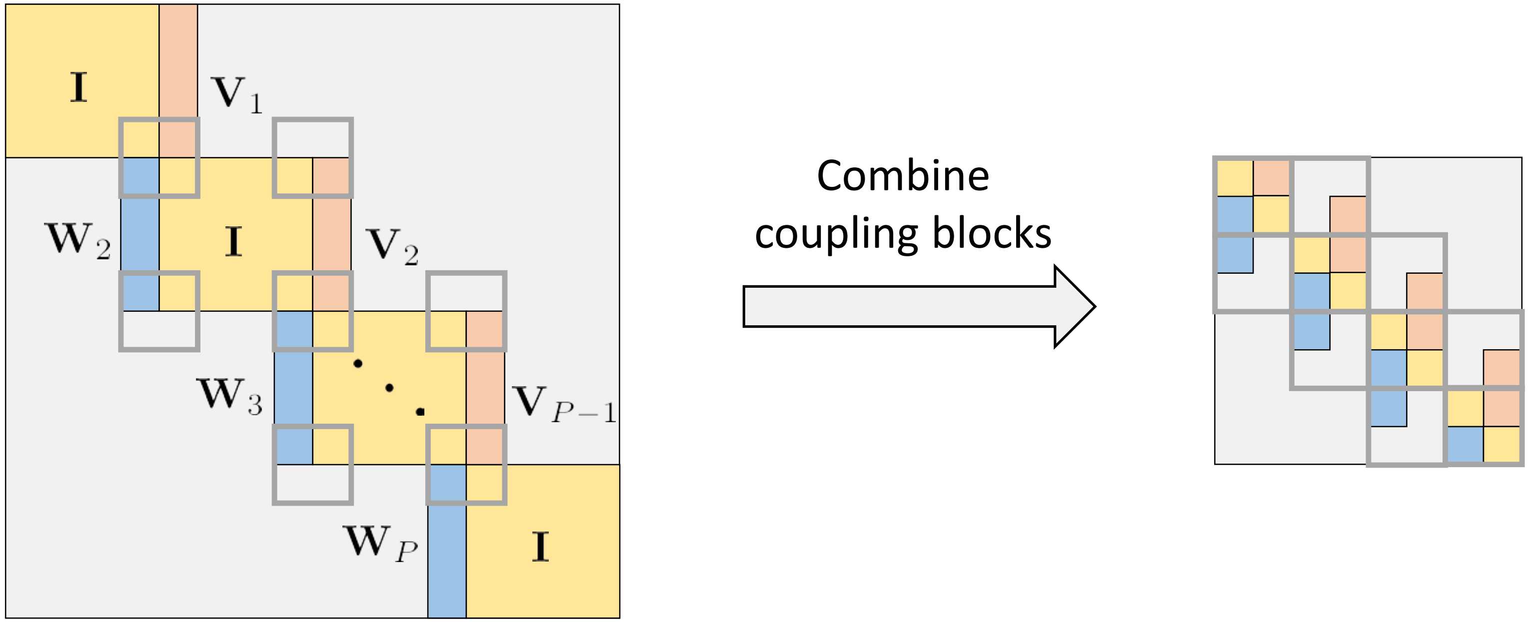
SaP::GPU: Algorithmic Details

- Solving $\mathbf{Dg}=\mathbf{b}$
- Reduced to solving P independent (banded dense) linear systems.
- Map these systems to P blocks on GPU.
- Apply classical LU (or UL) methods to each sub-system.



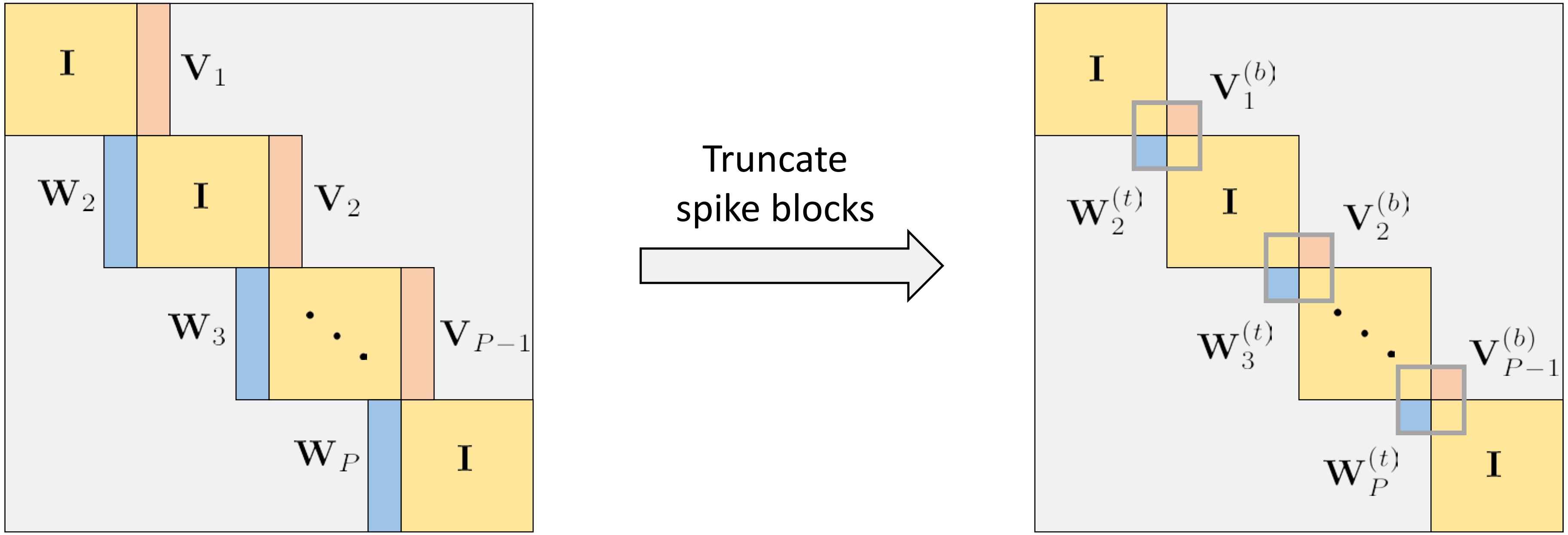
SaP::GPU: Algorithmic Details

- Solving $\mathbf{Sx}=\mathbf{g}$ (full SPIKE)
- Combine all coupling blocks into a *reduced* matrix
- (Recursively) solve the reduced system
- Recover solution from reduced solution



SaP::GPU: Algorithmic Details

- Solving $\mathbf{S}\mathbf{x}=\mathbf{g}$ (truncated SPIKE)
- Justified for **diagonally dominant** systems only.
- All spike blocks $\mathbf{W}$ and $\mathbf{V}$ are approximated by their top and bottom parts, respectively.
- Results in a decoupling of the reduced matrix into (P-1) small independent systems ($2K$ x $2K$).



SaP::GPU: Example

- SaP::GPU has been used to dramatically decrease the execution time for the linear solve in flexible multibody dynamics

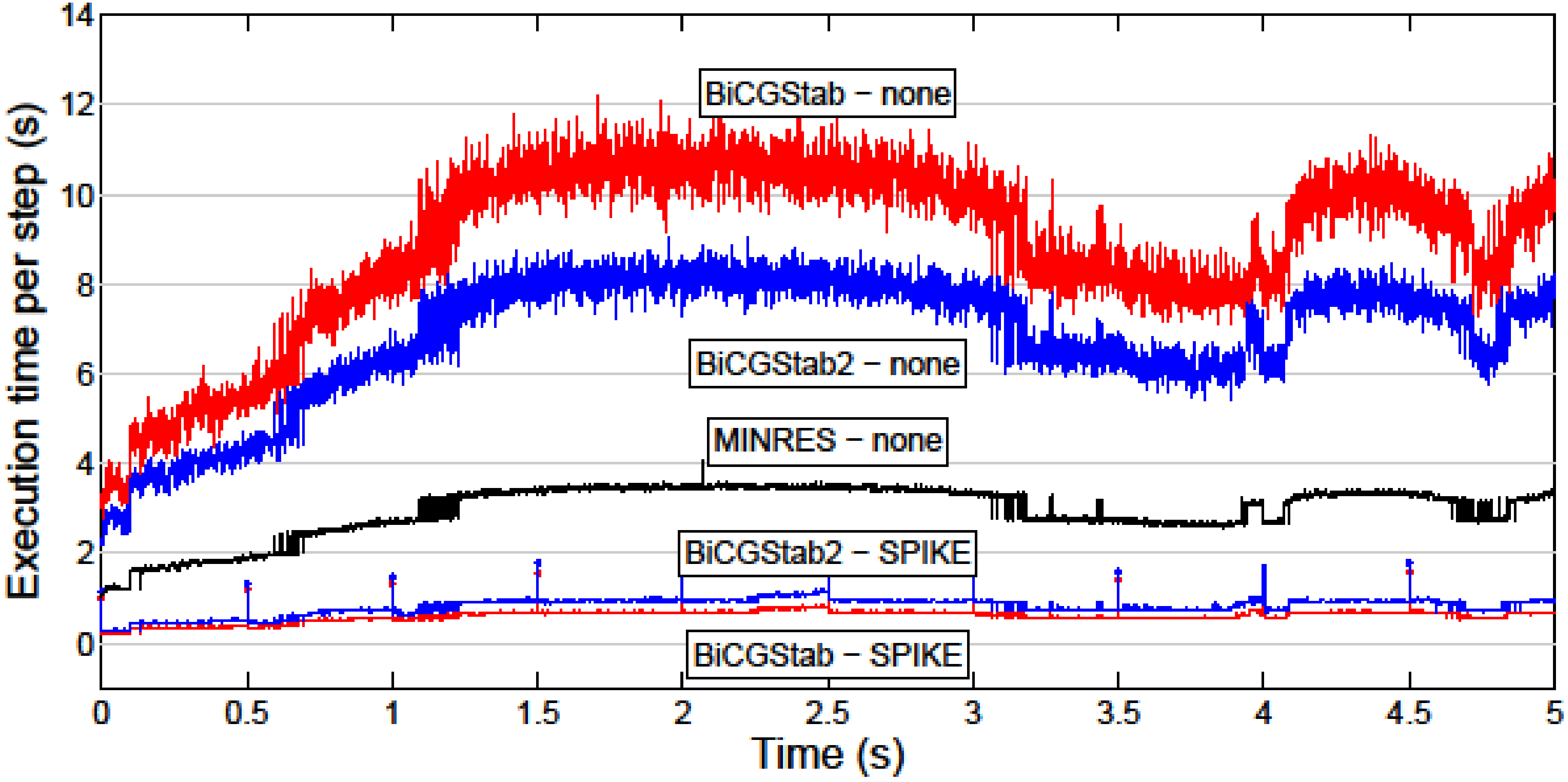
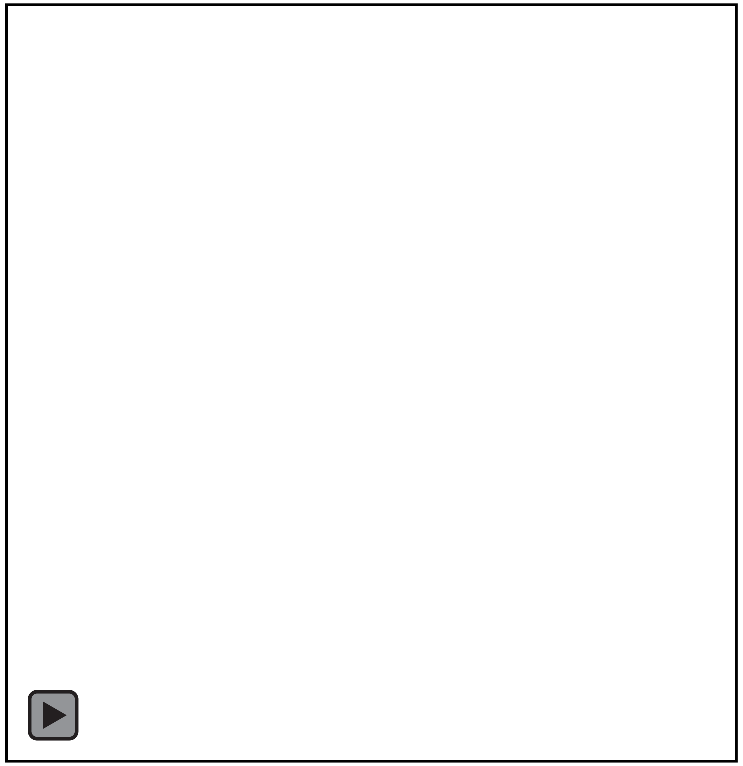


Fig. 8: Execution time for several linear solvers.

Preconditioned PDIP (P-PDIP) Results

- Adding preconditioning to the calculation of the search direction drastically improves the computation time



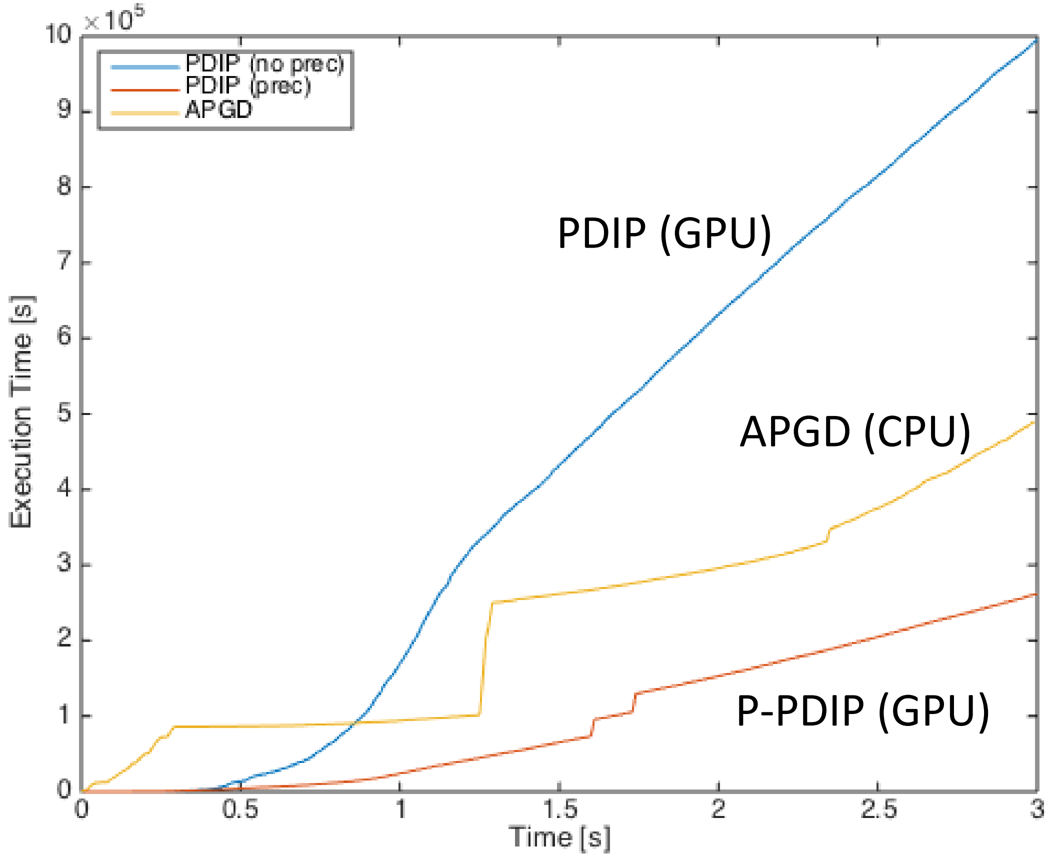


Fig. 9: Execution time for the filling test.

V. Numerical Results

Convergence Criteria: Filling Problem

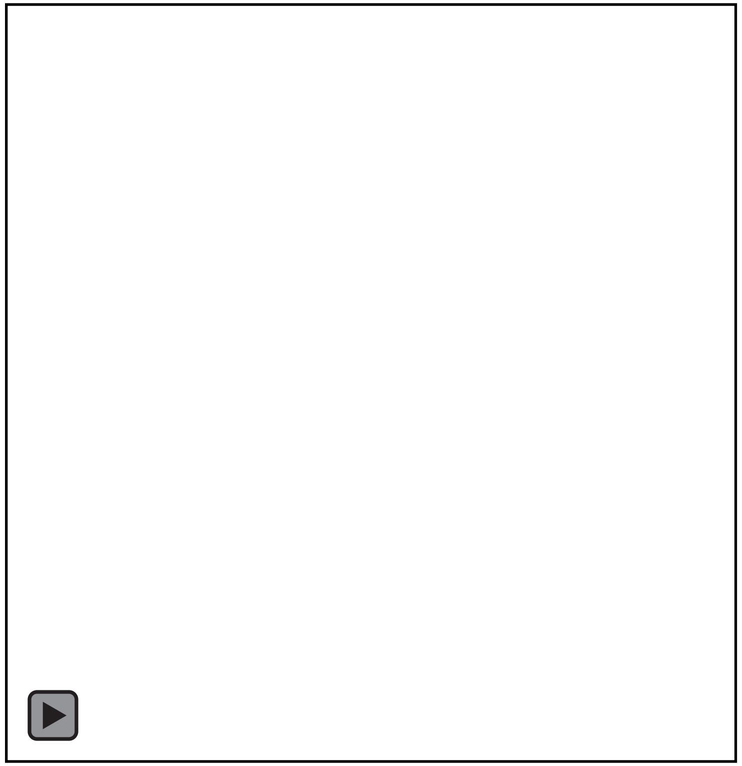


Fig. 10: Filling test.

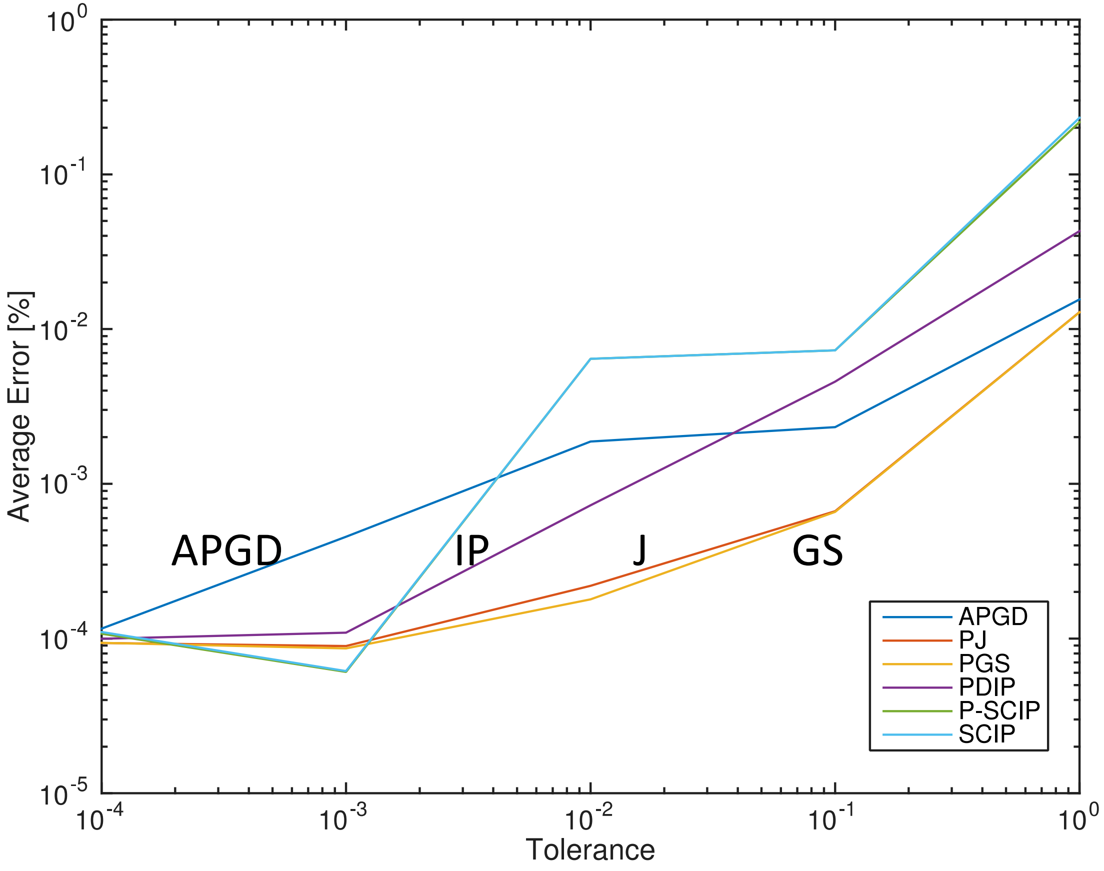
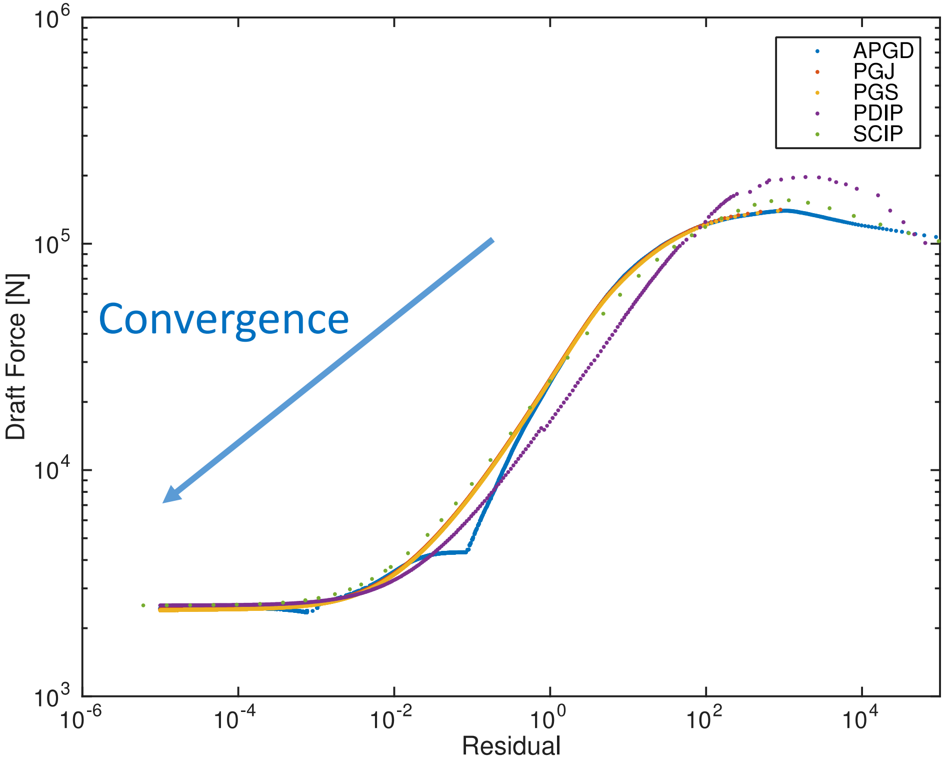
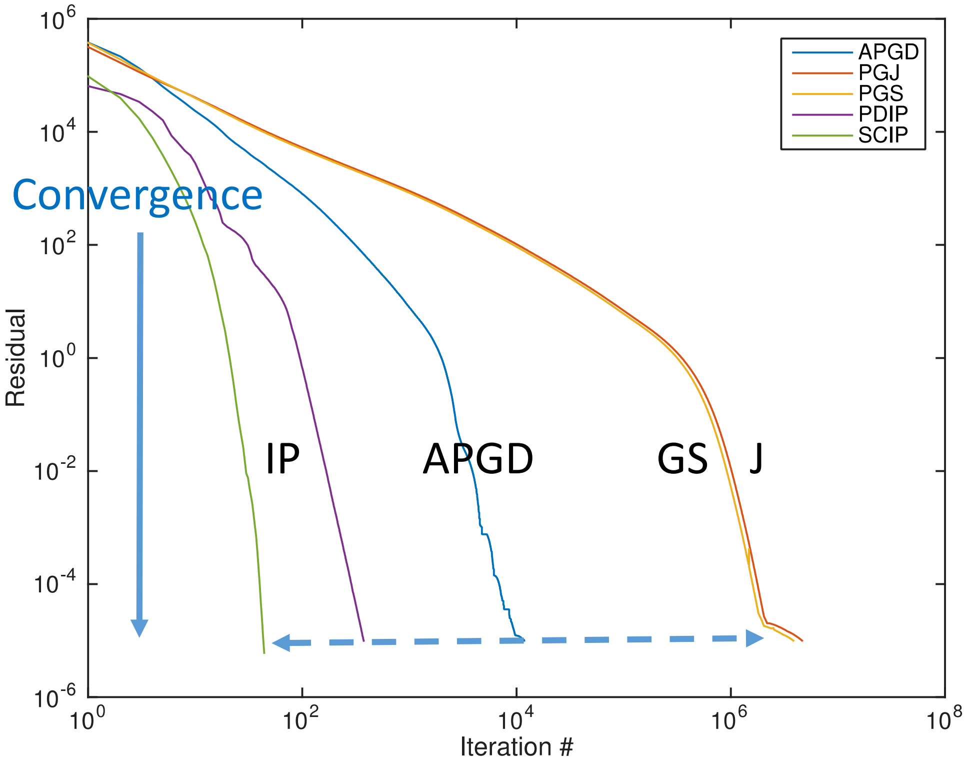


Fig. 11: Average error as a function of the filling test.

Solver Comparison: Bulldozing Problem





Solver Comparison

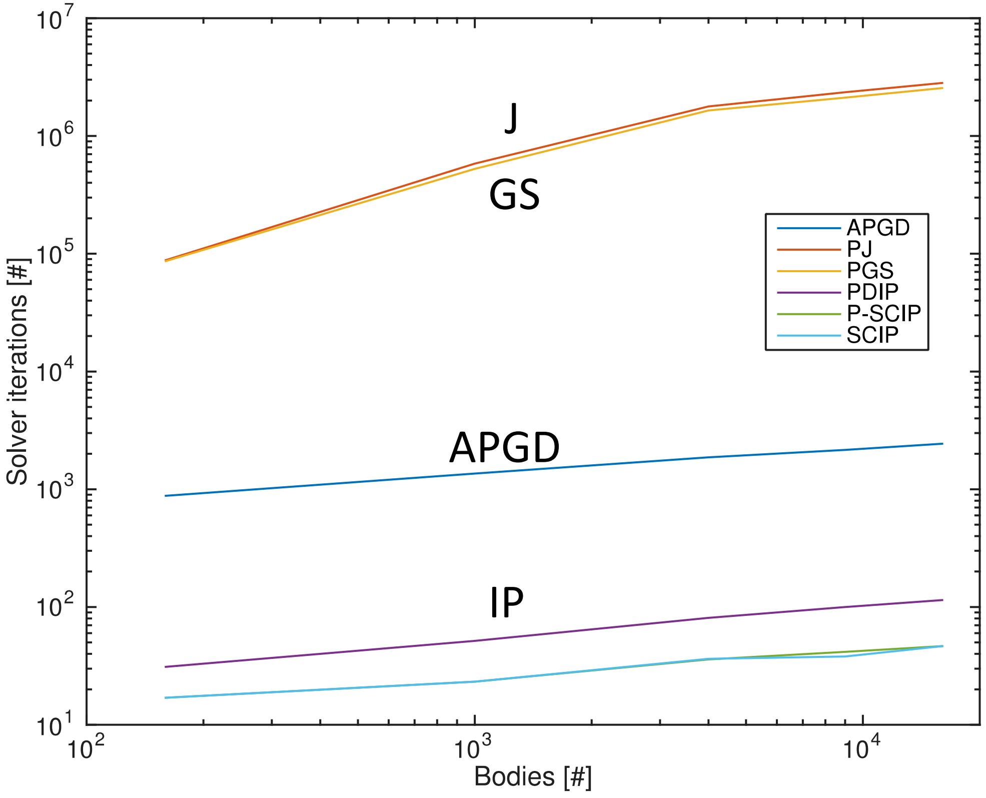


Fig. 13: Iterations as a function of the number of bodies.

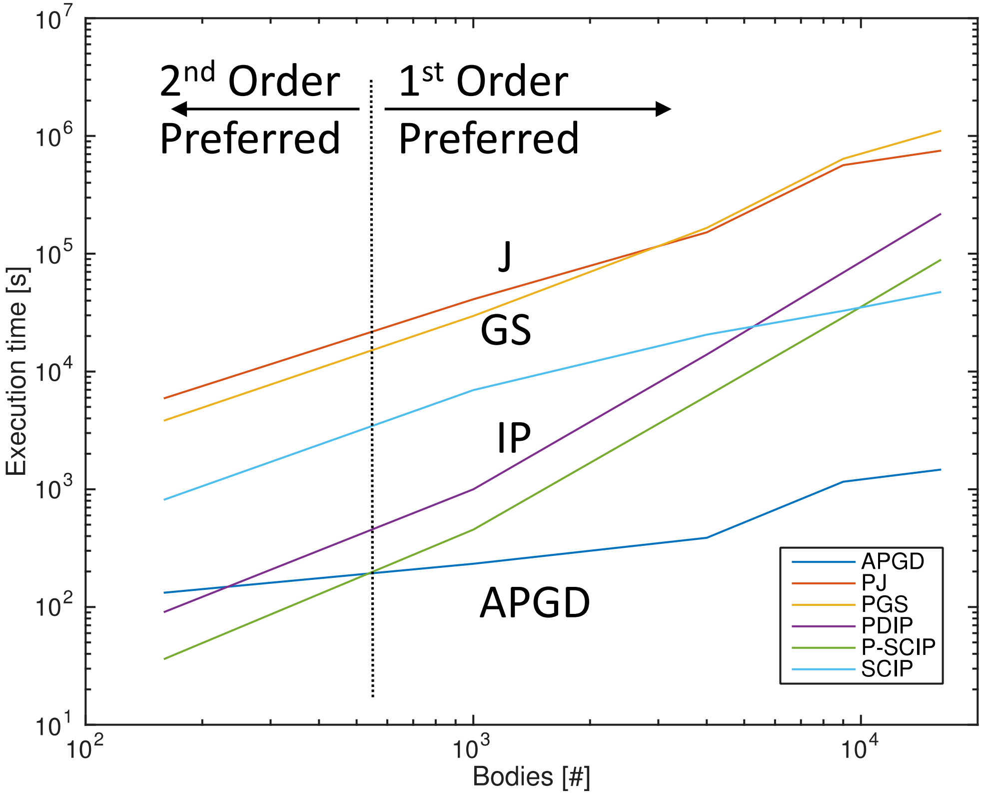


Fig. 14: Execution time as a function of the number of bodies.

VI. Conclusions

Summary

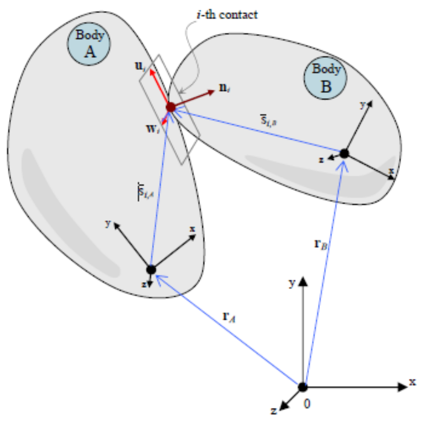
- Interior point methods require much fewer iterations than gradient descent methods, but each iteration is much more computationally expensive
- Using the GPU can result in 8-10X decreases in computational time
- For problems with less than 500 bodies, using second-order methods result in a 4X speedup over first-order methods (32-40X overall)
- Future work may be focused on investigating improvements to the interior point solvers
  - They might prove useful in robotics and real-time gaming scenarios where their speed of convergence makes them attractive, still may not be real time
  - For large-scale terramechanics problems, first-order method such as APGD proves to be the most efficient which leaves us at status quo

Research Productivity

1) Conference paper presented at European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS Congress 2016, Crete, Greece, June, 2016.

2) Journal paper "A comparison of numerical methods for solving rigid-body dynamics problems with frictional contact modeled via differential variational inequalities" submitted to the *J. Comp. Physics in June 2016*.

Physics of Terramechanics and Quantum Computing



$$\dot{\mathbf{q}} = \mathbf{L}(\mathbf{q})\mathbf{v}$$

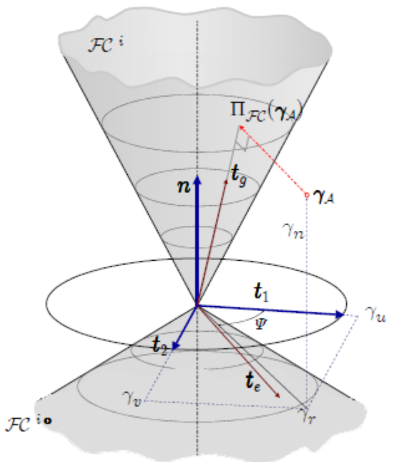
$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \mathbf{f}(t,\mathbf{q},\mathbf{v}) - \mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q},t)\lambda + \sum_{i\in\mathcal{A}(\mathbf{q},\delta)} \underbrace{\left(\widehat{\gamma}_{i,n}\,\mathbf{D}_{i,n} + \widehat{\gamma}_{i,u}\,\mathbf{D}_{i,u} + \widehat{\gamma}_{i,w}\,\mathbf{D}_{i,w}\right)}_{\text{Frictional Contact Force}}$$

$$0 = \mathbf{g}(\mathbf{q},t)$$

$$i \in \mathcal{A}(\mathbf{q}(t),\delta) \quad : \quad \begin{cases} 0 \le \Phi_i(\mathbf{q}) \quad \perp \quad \widehat{\gamma}_{i,n} \ge 0 \\ (\widehat{\gamma}_{i,u}, \widehat{\gamma}_{i,w}) = \underset{\sqrt{(\bar{\gamma}_u^i)^2 + (\bar{\gamma}_w^i)^2} \le \mu_i \widehat{\gamma}_{i,n}}{\operatorname{argmin}} \quad \mathbf{v}^T \cdot \left(\bar{\gamma}_u^i\,\mathbf{D}_{i,u} + \bar{\gamma}_w^i\,\mathbf{D}_{i,w}\right) \end{cases}$$

Complementarity Condition

Friction Dissipation Energy



Mobility Problem reduces to:
Quadratic **C**onstrained **C**ontinuous **O**ptimization Problem

Note that Quantum Computer can solve:
Quadratic **U**nconstrained **B**inary **O**ptimization Problem

Quantum Annealing for Mobility Studies:
Generation of GO/NO-GO Maps via Quantum-Assisted Machine Learning

University of Wisconsin – Madison

NASA

U.S. Army TARDEC

Objective and project goals

- Objective: provide proof-of-concept approach to solving ground vehicle mobility-related problems on emerging quantum computing machines

- Target problem: Generation of Go/NoGo maps

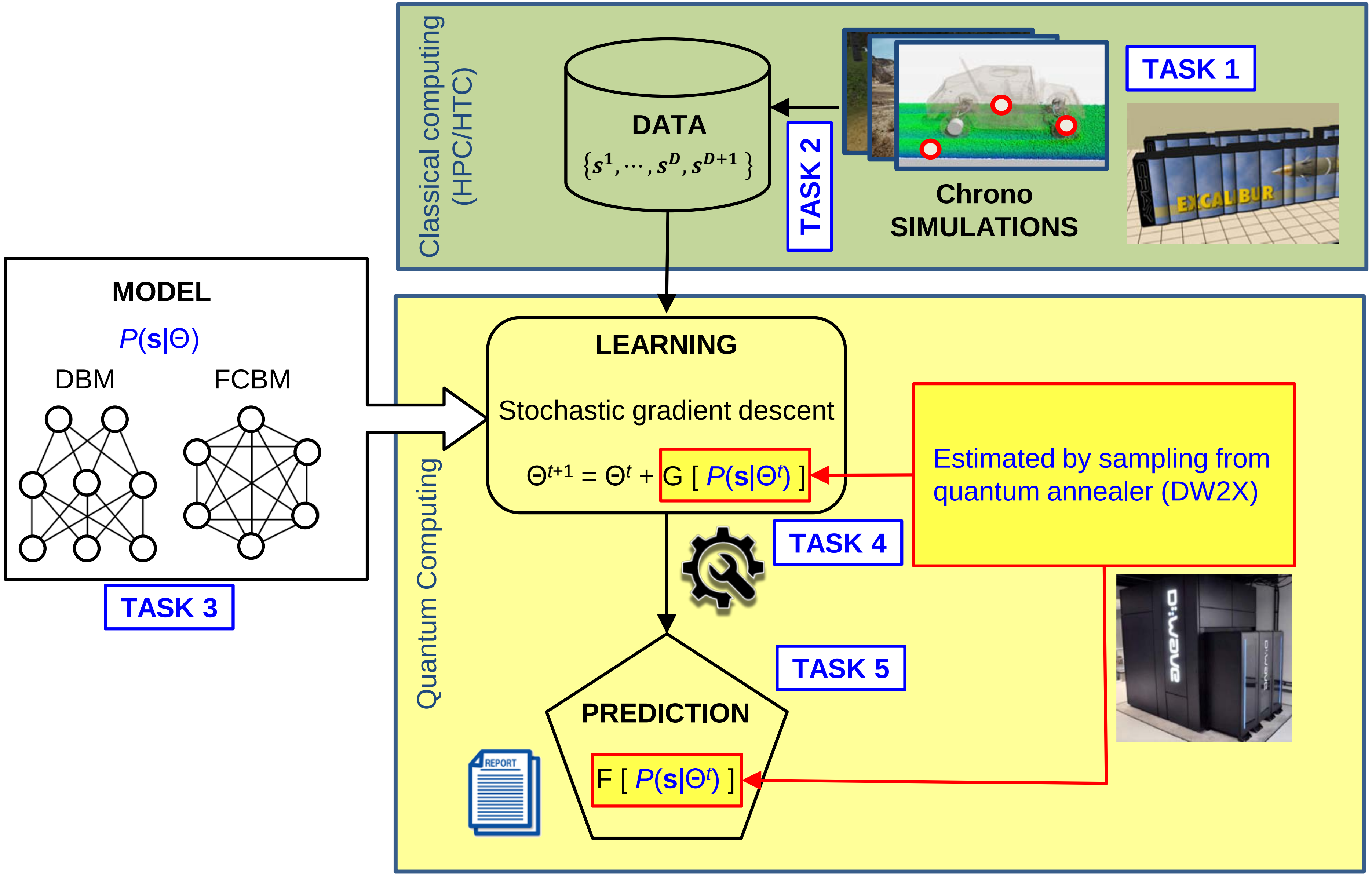
- Project goals:
  1. Cast the Go/NoGo problem as a machine learning problem
  2. Identify appropriate problem variables (input data and labels)
  3. Develop the quantum-assisted learning algorithm
  4. Generate the training and validation data sets using classical computing
  5. Asses performance of QC-assisted machine learning for the Go/NoGo problem
  6. Identify other vehicle-mobility problems that can benefit from QC and QC-assisted machine learning

Quantum computing: Opportunities and challenges

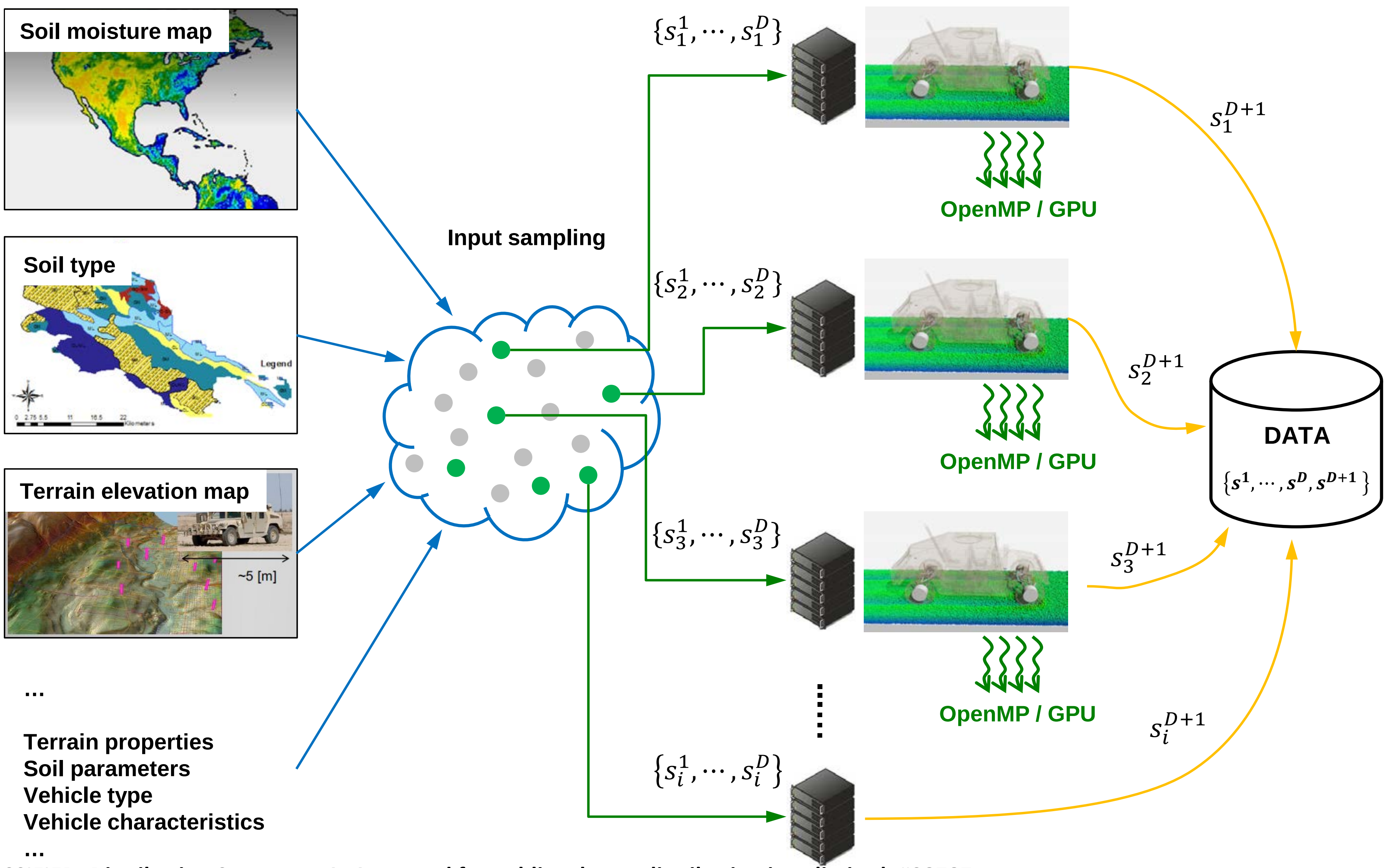
- Opportunities
  - QC holds the potential for significant speedups in high-dimensional cases
  - Machine learning is a success story with the D-Wave machines
  - QC particularly appropriate for unsupervised learning
    - Traditional sampling techniques (e.g., MCMC) are intractable on classical computers
    - Quantum annealing can sample more efficiently and from more complex probabilistic models

- Challenges (D-Wave architecture)
  - Special-purpose architecture that can encode a problems with quadratic energy functions (pairwise interactions between qubits), aimed to solve optimization problems or to speed up machine learning tasks involving Gibbs sampling
  - Hardware limitations
    - D-Wave 2X: 1152 qubits, 3000 couplers
    - Problems must be mapped onto the underlying Chimera graph architecture
    - Problem size limited to (worst-case) $\sim\sqrt{1152}$
  - Precision limitations

Data processing pipeline: Hybrid quantum-classical approach



Generation of training set (HPC/HTC)



Project plan

- Main objectives/tasks:
  - Task 1: Selection of most important input variables informed by Chrono simulations.
  - Task 2: Suitable generation of a training set that ensure proper coverage of the input space.
  - Task 3: Selection of probabilistic model that balances between number of variables and connectivity.
  - Task 4: Development, execution, and fine-tuning of learning algorithms.
  - Task 5: Analysis of results and report recommendations for future developments.

- Deliverables:
  1. Complete set of Chrono models and simulations used to generate the machine learning training set.
  2. Tuned learning algorithm and QC implementation for training complex generative models that can answer mobility-related questions.
  3. Demonstration of capability of generative models to both produce go/no-go decisions given the relevant variables; and reconstruct all the relevant variables from a subset of them.

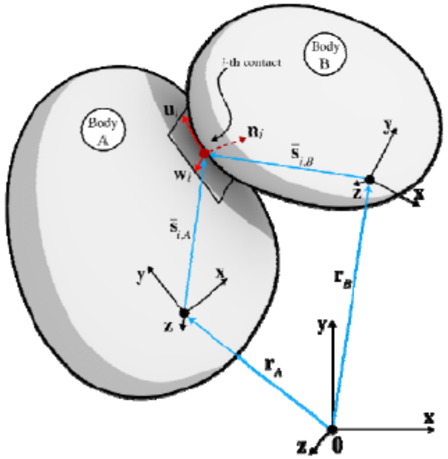
Fast numerical algorithms for high-fidelity simulation of terramechanics

Shravan Veerapaneni, University of Michigan
Eduardo Corona, University of Michigan
Paramsothy Jayakumar, U.S. Army TARDEC

Problem: Non-smooth Rigid Body Dynamics

Nonsmooth Rigid Body Dynamics

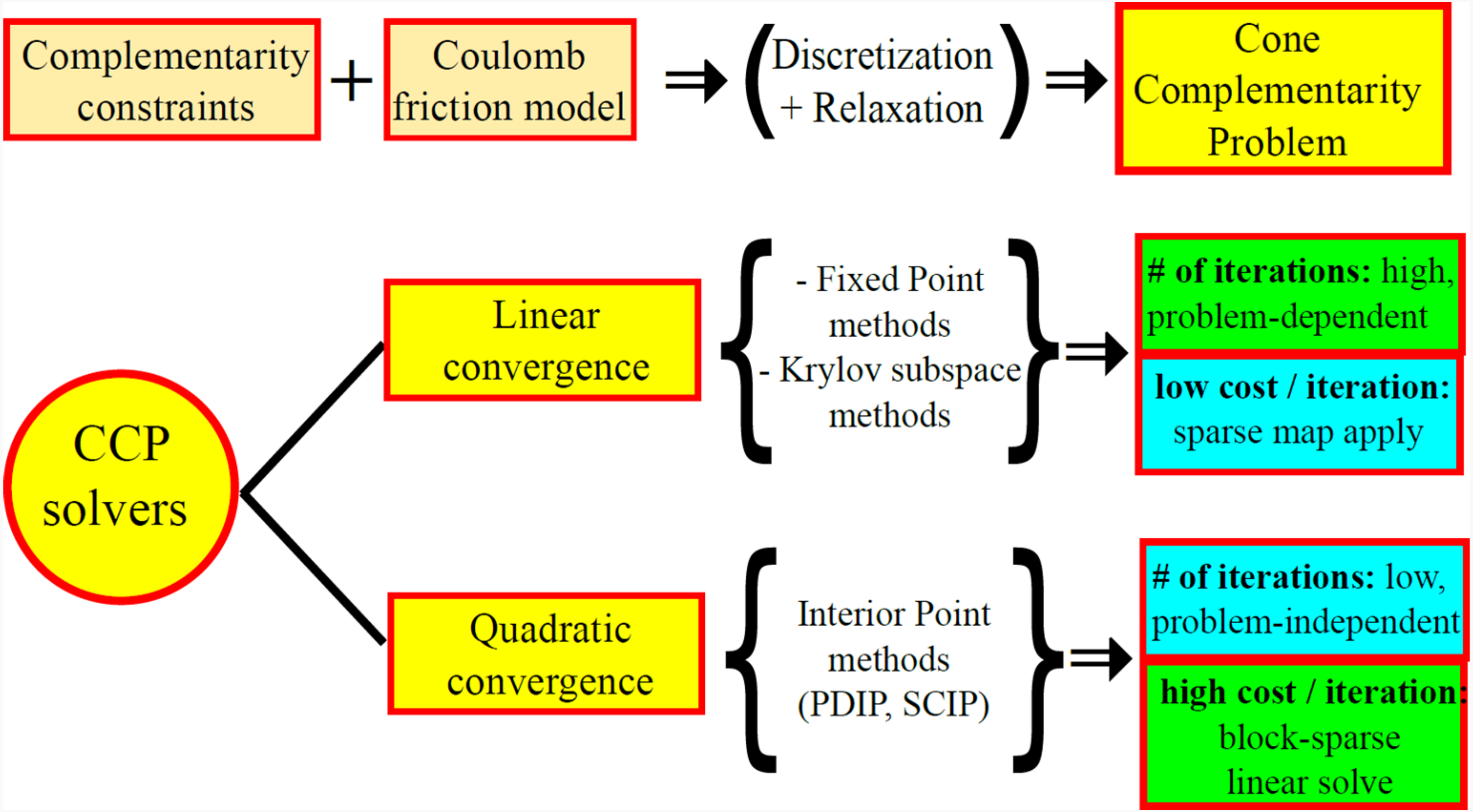
- **Penalty**
- **Complementarity**



Penalty	Complementarity
Pros: - Computationally cheap and easy to implement. - Flexible and adaptable	**Pros:** - Does not introduce stiffness - Timestep size is dictated by collision frequency and accuracy
Cons: - Introduces high stiffness - Small timesteps	**Cons:** - DVI problem discretization - Nonlinear optimization solve

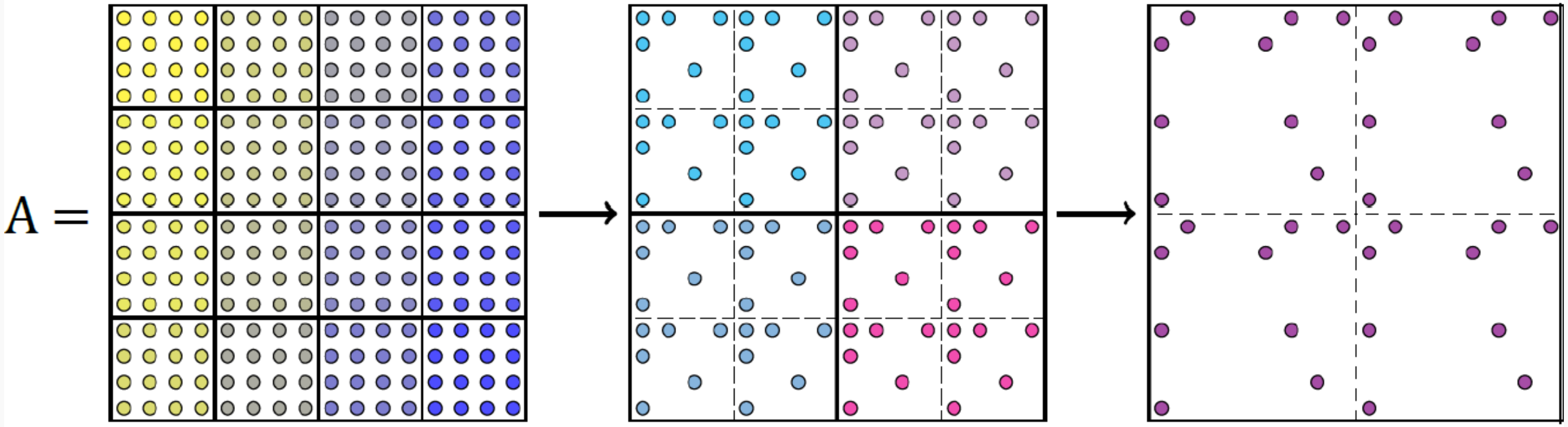
Project Goal: numerical solvers to accelerate and increase efficiency and robustness of **complementarity** approach

Complementarity approach: optimization solvers



Accelerating Interior Point Methods: Fast Direct Solvers

- **IP iteration -** Newton step solve Az = -r, with A a $5N_{col} \times 5N_{col}$, N_{col} = colliding pairs
- **Tensor train decomposition:** hierarchical **compression** and **inversion**



For a large class of structured matrices, we have:

- **Storage and inverse compression** - $o(N_{col})$ (sublinear)
  Solve - $O(N_{col} \log N_{col})$

- **Does this behavior extend to the Newton matrix?**
- **Use as direct solver or as low cost preconditioner?**
- **Can we update quickly if active set changes?**