Code Generation for Receding Horizon Control

Jacob Mattingley

joint work with Stephen Boyd and Yang Wang

Electrical Engineering Department, Stanford University

Overview

- convex optimization for RHC
- modeling tools and solvers
- automatic code generation with CVXGEN
- detailed example

RHC optimization problem

$$
\begin{array}{ll}
\text{minimize} & \frac{1}{T+1}\sum_{\tau=t}^{t+T}\ell_\tau(x_\tau,u_\tau) \\
\text{subject to} & x_{\tau+1} = A_\tau x_\tau + B_\tau u_\tau + c_\tau, \\
& (x_\tau,u_\tau)\in\mathcal{C}_\tau, \quad \tau = t,\ldots,t+T \\
& x_t = \hat{x}_{t|t}
\end{array}
$$

- optimization variables: state $x_t,\ldots,x_{t+T+1}$; input $u_t,\ldots,u_{t+T}$
- problem data: A_τ, B_τ, c_τ, $\mathcal{C}_\tau$, ℓ_τ, current state estimate $\hat{x}_{t|t}$
- assume convex constraint sets $\mathcal{C}_t$, objective functions ℓ_t

Solving the RHC optimization problem

- QP with equality constraints: analytic solution
- small number of constraints, very low dimension: explicit MPC
- otherwise: iterative method (first order, active set, interior point)

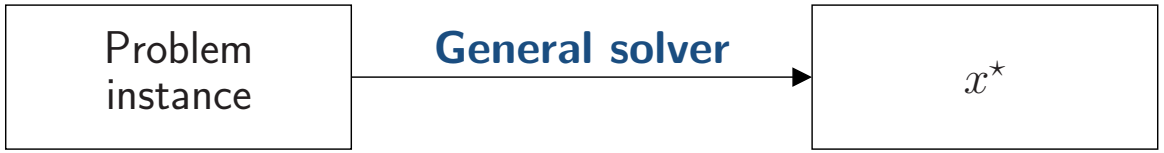
Embedded convex optimization

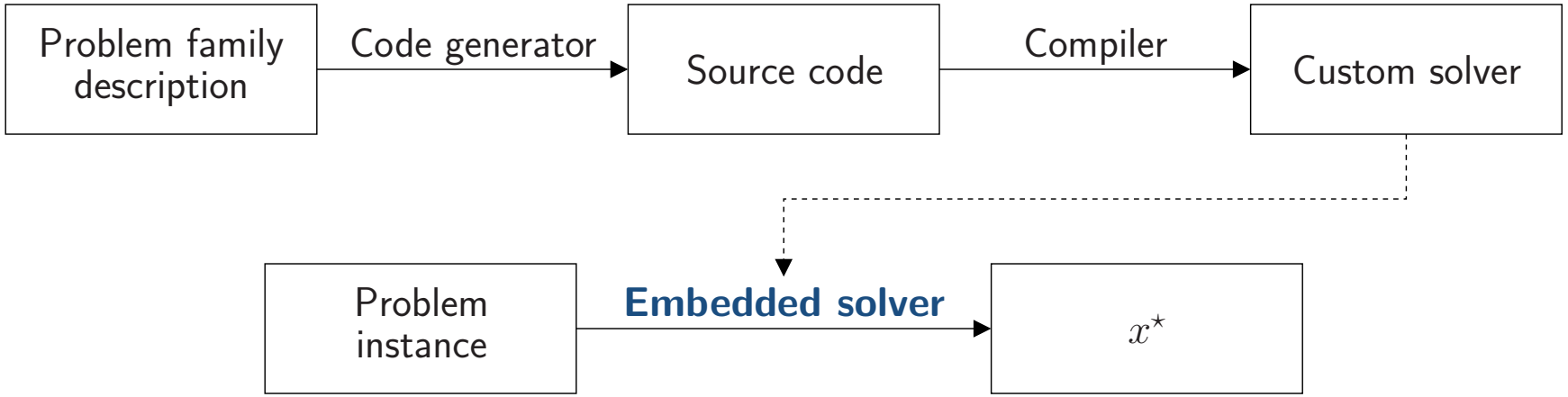
- small, fixed problems
- millisecond, even microsecond, time scales
- online applications, possibly with real-time deadlines
- **includes many RHC applications**

Modeling tools and solvers

- parser/solvers
  - automate modeling, verification, transformations
  - convenient in the design phase
  - but, often too slow for online applications
- handwritten solvers
  - can be much faster
  - but, time-consuming and difficult to write
- code generators
  - modeling, verification, transformations, code generation **offline**
  - extremely fast solution **online**
  - (code generators not a new idea)

Modeling tools and solvers





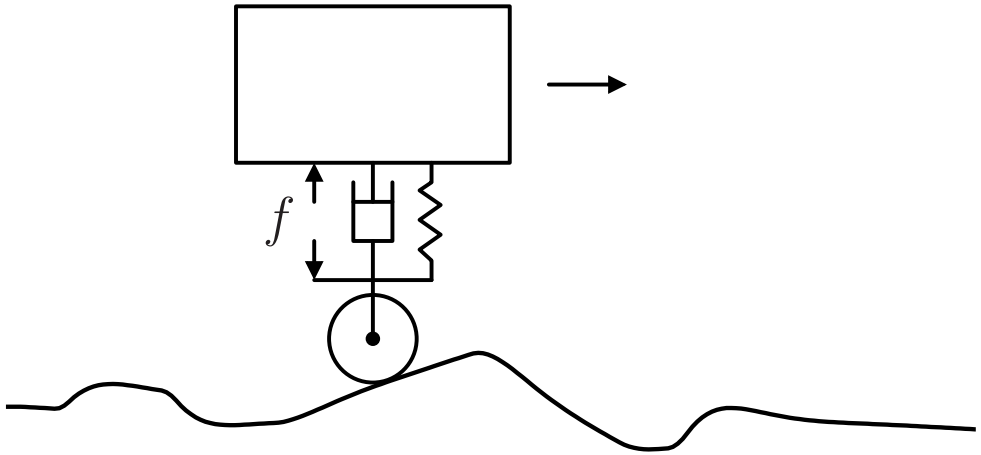
Automatic code generation

- **offline**
  - decide transformations, permutations, memory usage
  - exploit problem structure, sparsity
  - generate very explicit code
  - use optimizing compiler
- **online**
  - use highly optimized solver, for single problem family
  - typical speed-up over efficient general solver: 100–10,000 $\times$

CVXGEN

- Mattingley, Boyd
- targets small QPs, including many RHC problems
- Mehrotra predictor-corrector, primal-dual, interior-point method
- generates library-free C code
- sometimes works surprisingly well: up to 10,000 $\times$ faster than CVX+SDPT3

Example: Active suspension control



- mass-spring-damper model of vehicle suspension, with actuator control f
- minimize ride roughness, subject to various constraints
- use embedded convex optimization for RHC, via CVXGEN

Suspension: Problem formulation

$$
\begin{array}{ll}
\text{minimize} & \frac{1}{T+1}\sum_{\tau=t}^{t+T}\left(a_\tau^2 + \rho u_\tau^2 + \mu(y_\tau - \hat{h}_{\tau|t})^2\right) \\
\text{subject to} & x_{\tau+1} = Ax_\tau + bu_\tau + \hat{v}_{\tau|t}, \quad y_\tau = c^T x_\tau, \quad a_\tau = d^T x_\tau + gu_\tau + \hat{w}_{\tau|t}, \\
& E^{\min} \leq y_\tau - \hat{h}_{\tau|t} \leq E^{\max}, \quad F^{\min} \leq u_\tau \leq F^{\max}, \quad \tau = t, \ldots t+T \\
& x_{t+T+1} = 0, \quad x_t = \hat{x}_{t|t},
\end{array}
$$

- model coefficients $A \in \mathbf{R}^{n\times n}$, $b \in \mathbf{R}^n$; exogenous terrain force $v_t \in \mathbf{R}^n$
- variables: vehicle dynamic state $x_t \in \mathbf{R}^n$; height y_t; vertical acceleration a_t
- constraints: extension limits E; force limits F
- objective: minimize ride roughness

Suspension: CVXGEN modeling

```
dimensions
  n = 2; T = 20
end

parameters
  A (n,n); B (n,1)
  C (1,n); D

  h[t], t=0..T+1
  v[t] (n), t=0..T
  w[t], t=0..T

  rho positive; mu positive
  Fmin; Fmax;  Emin; Emax
  x[0] (n)
end
```

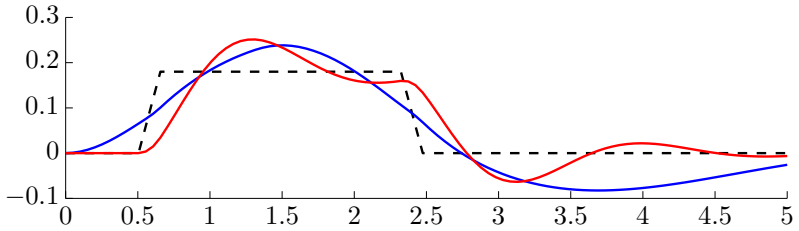
```
variables
  u[t], t=0..T
  x[t] (2), t=1..T+1
  a[t], t=0..T
end

minimize
  sum[t=0..T](square(a[t]) + rho*square(u[t])
   + mu*square(x[t][1] - h[t]))
subject to
  x[t+1] == A*x[t] + B*u[t] + v[t], t=0..T
  a[t] == C*x[t] + D*u[t] + w[t], t=0..T
  Fmin <= u[t] <= Fmax, t=0..T
  Emin <= x[t][1] - h[t] <= Emax, t=1..T
end
```

Suspension: CVXGEN results

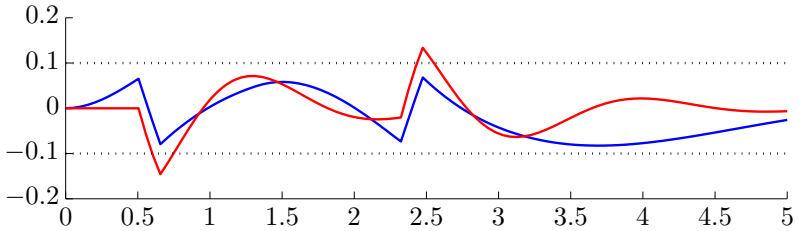
- 105 variables, 170 constraints
- 13.2 s to generate at cvxgen.com
- 440 kB of code
- 110 μs to solve on Intel i7 (2.5 s in CVX+SDPT3)

Suspension: Simulation

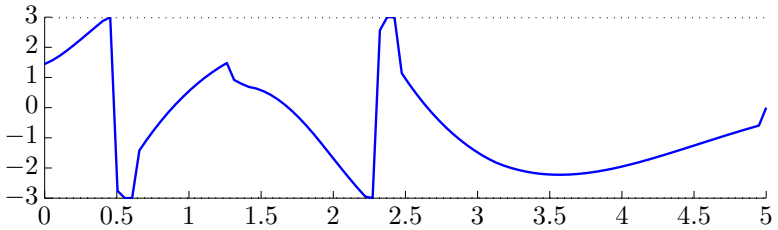
vehicle height
(active blue, passive red)



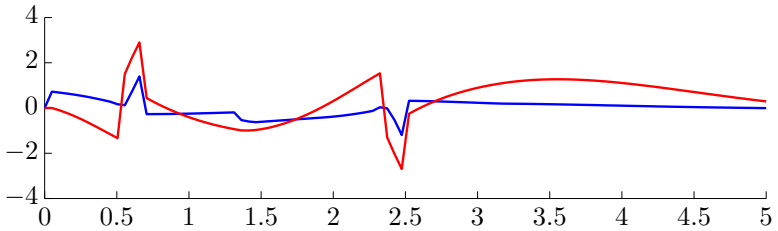
suspension extension



suspension force



vertical acceleration



CVXGEN solver speed

	`proc_speed`	`storage`	`suspension`
Variables	279	153	104
Constraints	465	357	165
CVX, Intel i7 (ms)	4190	1290	2570
CVXGEN, Intel i7 (ms)	0.85	0.36	0.11
Solve rate, low power Atom	130 Hz	250 Hz	1 kHz

Conclusions

- RHC gives good performance, while respecting constraints
- code generation makes it easy to write fast solvers
- many applications can benefit from RHC, especially with code generation