Simple and Certifiable QP Algorithms for Embedded Linear MPC

Alberto Bemporad, Panagiotis Patrinos

Institutions Markets Technologies
IMT
INSTITUTE FOR ADVANCED STUDIES LUCCA

DYSCO
Research Unit

Dynamical Systems, Control, and Optimization

NMPC'12

Noordwijkerhout, The Netherlands, August 23, 2012

Outline

- Motivating applications (MPC for aerospace industry)
- QP solvers for embedded MPC
- A new "accelerated dual gradient projection" algorithm
- Numerical examples
- Conclusions

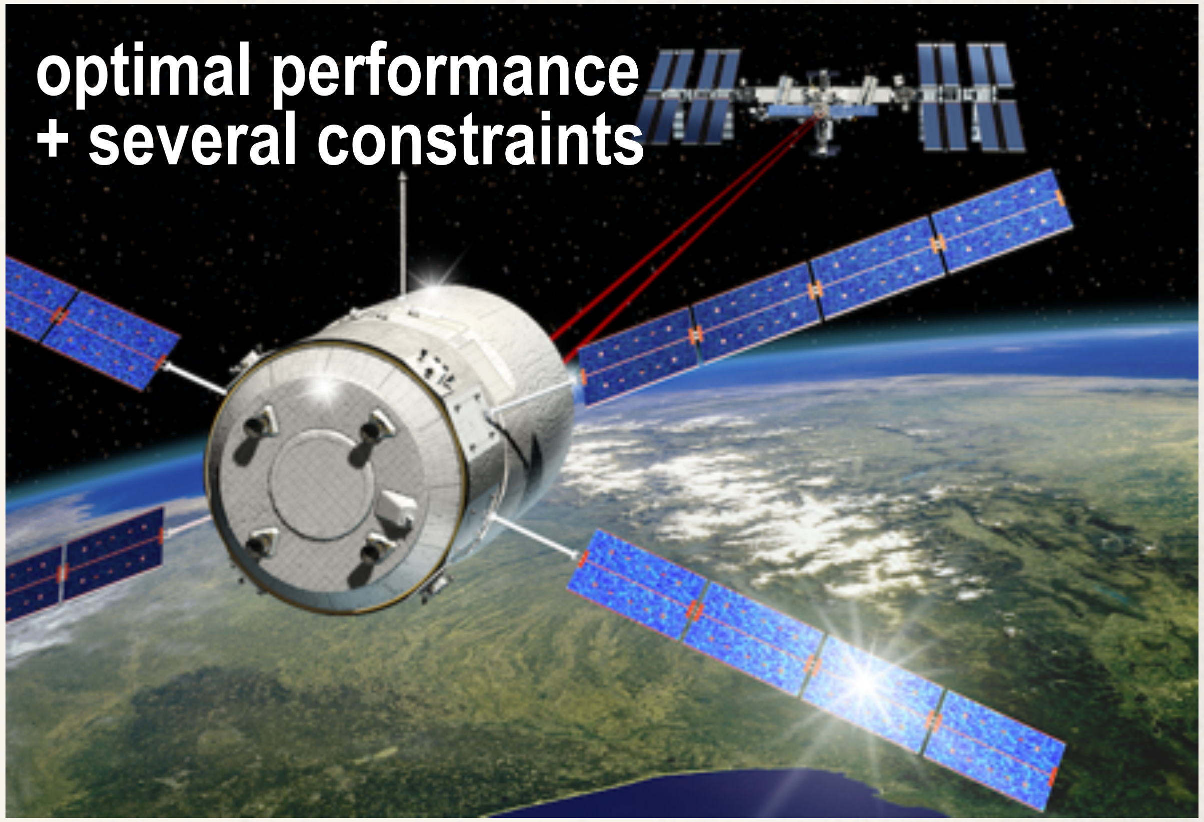
ESA-ESTEC, Noordwijk, The Netherlands

MPC for aero applications

- Plenty of potential MPC applications in aerospace industry !

  - Rendezvous and docking
  - Satellite formation flying
  - Unmanned aerial vehicles
  - Planetary rover locomotion (traction, guidance)
  - EDL (Entry Descent and Landing)
  - Ascent closed-loop guidance
  - In orbit servicing
  - ... and many others !

  http://www.esa.int/SPECIALS/Technology/



ATV Johannes Kepler docks with International Space Station

- Only a few research groups active

  - Richards, Trodden (Bristol, UK)
  - Maciejowski, Hartley (Cambridge, UK)
  - Balas, Borrelli, Keviczky (Minnesota)
  - ...
  - Murray, Dunbar (Caltech)
  - How, Breger, Tillerson (MIT)
  - Bemporad, Pascucci, Rocchi (IMT Lucca)

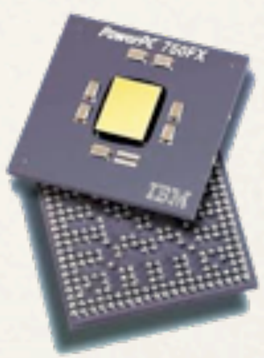
- Aerospace industry has very demanding **certification standards**

MPC for aero applications

- ESA funded project “**ORCSAT**” (2009-2011)
  - Main emphasis on **real-time implementation** capabilities of MPC (architecture, automatic generation of C and HDL code, timing)
  - Applications: orbit synchronization, impulsive hopping
  - New MATLAB Toolbox **MPCTOOL**, adding many new features on top of MPC Toolbox (such as LTV-MPC based on LP/QP)

AT697F processor not good, co-processor is needed

FPGA implementation from EML code not good

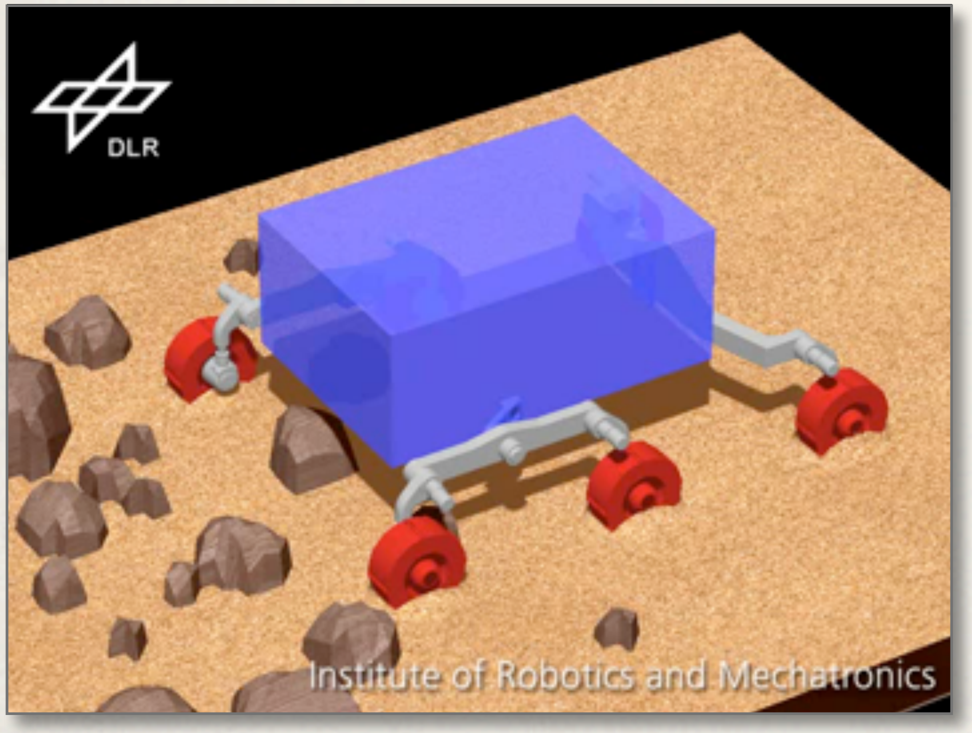


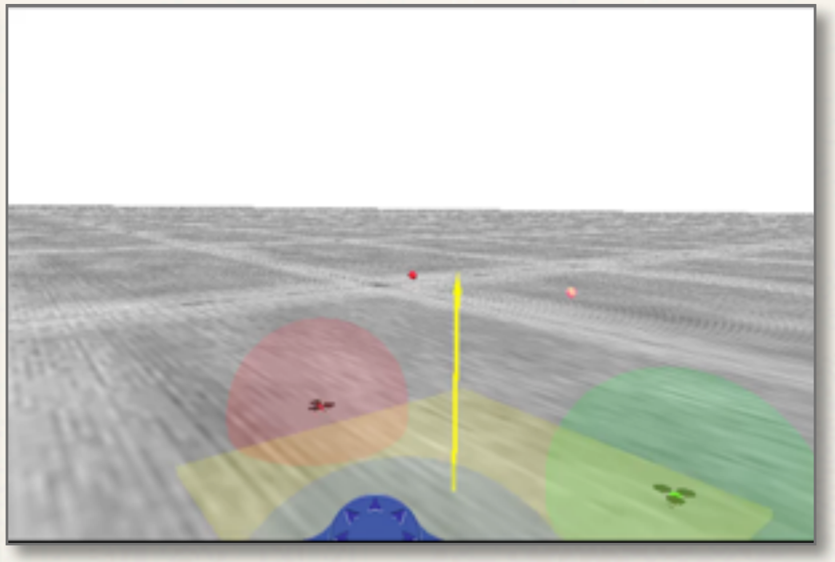
LTV-MPC implementation required PowerPC750FX

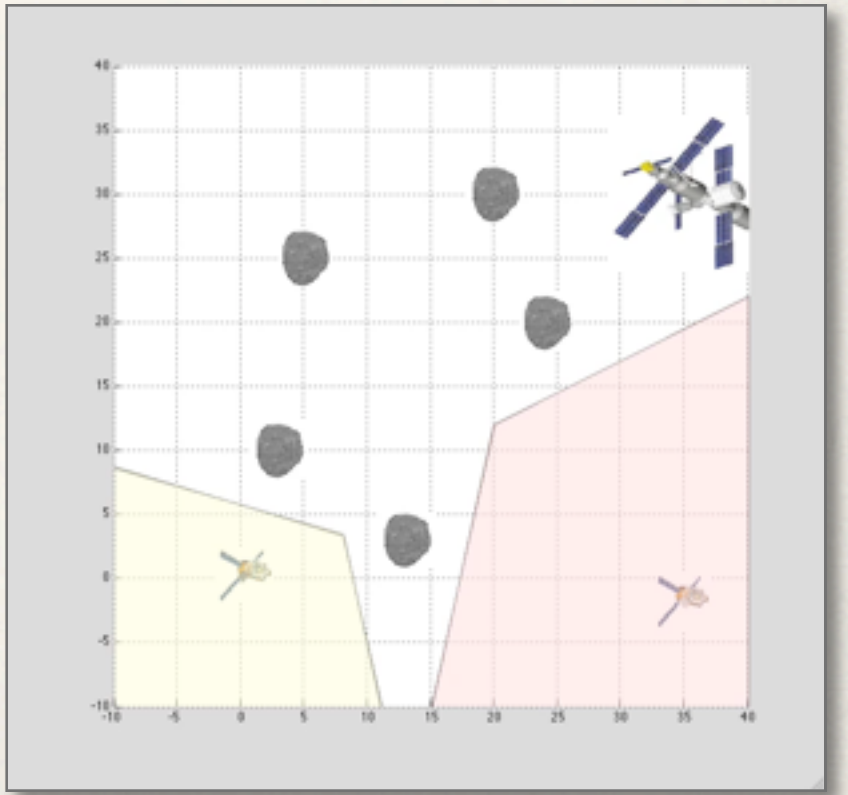
ORCSAT - On-line Reconfiguration Control System and Avionics Technologies

MPC for aero applications

- ESA funded project “**ROBMPC**” (2010-2011)
  - Main goal: explore MPC capabilities in new space apps
  - Applications: Cooperating UAVs, planetary rover locomotion (traction & guidance)
  - New MATLAB Toolbox **MPCSofT** for **LTV-MPC design**







ROBMPC - Robust MPC for Space Constrained Systems

(Bemporad, Rocchi, CDC 2011)

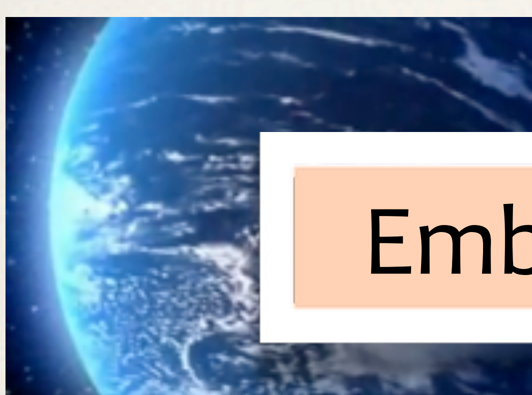
Embedded optimization in aerospace

- **PRISMA** project for autonomous formation flying (S. Persson, S. Veldman, P. Bodin, 2009)

http://www.lsespace.com/about-prisma.aspx

- Objective: **minimize fuel consumption** subject to keeping motion within a **box constraint** (solved by **linear programming**)

$$\begin{array}{ll} \min\limits_{dV(t)} & \sum_{t=0}^{\infty} \|dV(t)\|_1 \\ \text{s.t.} & \|r - y_r\|_1 \leq y_{\text{box}} \end{array}$$



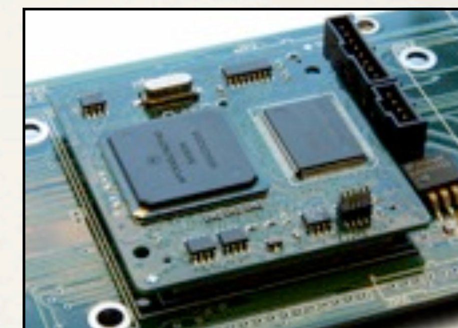
Embedded LP is **operational in space** !

How to make MPC technology really “fly” (in industry) ?

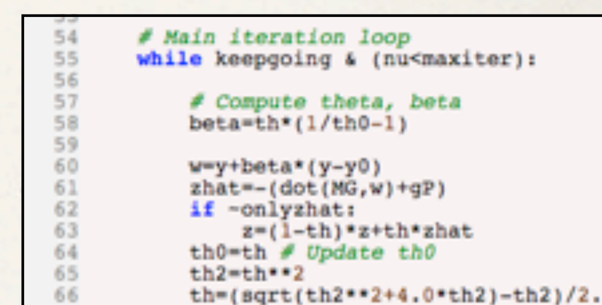
Requirements for Embedded MPC algorithms

- **Speed**: fast enough to provide a solution within short sampling intervals (such as 10-100 ms)

- Require **simple hardware** (microcontroller/FPGA) and little memory to store problem **data *and* code**



- **Code simple** enough to be **software-certifiable**



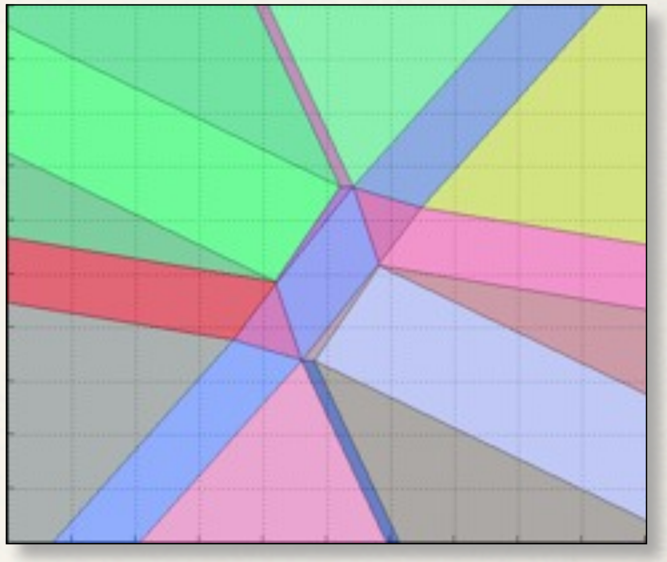
- Tightly estimate **worst-case execution time** to certify **hard real-time system** properties

Explicit model predictive control

- Presolve QP problem **off-line** for all states $x(t)$ in a given range via **multi-parametric programming**

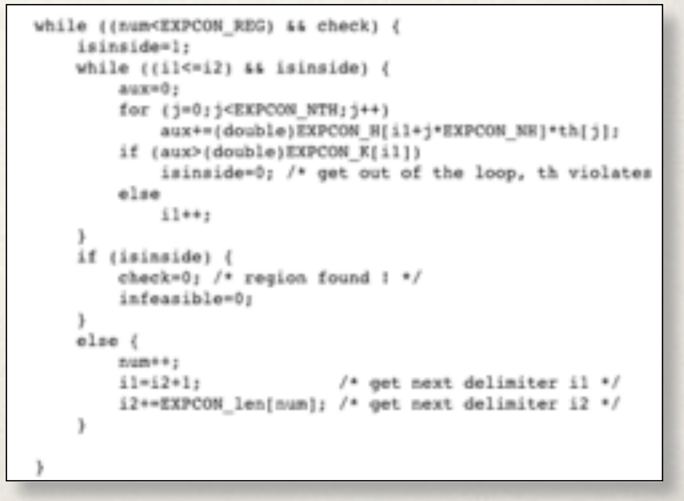
Contributors: Alessio, Baotic, Bemporad, Borrelli, Christophersen, De Doná, Dua, Filippi, Goodwin, Grancharova, Grieder, Johansen, Jones, Kerrigan, Kvasnica, Mayne, Maciejowski, Morari, Munoz de la Pena, Patrinos, Petersen, Pistikopulos, Rakovic, Sarimveis, Seron, Slupphaug, Spjøtvold, Tøndel, Zeilinger, ...

✓ Control law is **piecewise affine**
(and also continuous for linear MPC)



✓ Code **very simple**, **fast**, and **certifiable**

✖ Limited to **small problems**
(number of regions can grow exponentially with number of constraints)

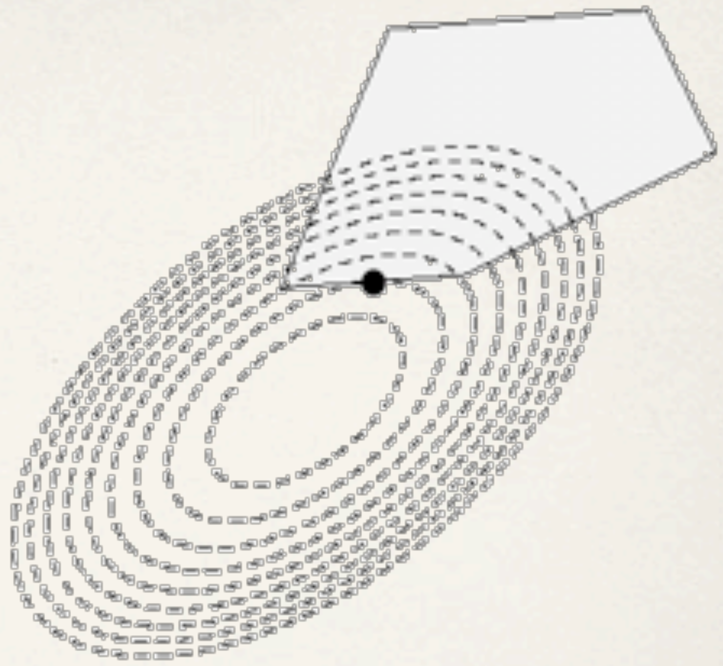


```
while ((num<EXPCON_REG) && check) {
     isinside=1;
     while ((i1<=i2) && isinside) {
          aux=0;
          for (j=0;j<EXPCON_NTH;j++)
               aux+=(double)EXPCON_H[i1+j*EXPCON_NH]*th[j];
          if (aux>(double)EXPCON_K[i1])
               isinside=0; /* get out of the loop, th violates
          else
               i1++;
     }
     if (isinside) {
          check=0; /* region found ! */
          infeasible=0;
     }
     else {
          num++;
          i1=i2+1;                /* get next delimiter i1 */
          i2+=EXPCON_len[num]; /* get next delimiter i2 */
     }

}
```

✖ Not applicable to **time-varying** models

On-line QP solution methods



- **active set** methods
  (best for **small/medium size** problems)

- **interior point** methods
  (best for **large-scale** problems)

- **gradient projection** methods (Goldstein, 1964) (Levitin & Poljak, 1965) (Nesterov, 1983)

- **conjugate gradient** methods

- **augmented Lagrangian** methods
  (and *alternating direction method of multipliers*)

Accelerated Dual Gradient Projection (GPAD)

Results based on (Patrinos, Bemporad, CDC 2012)
(Patrinos, Bemporad, journal version, submitted 2012)

Linear MPC - QP problem setup

- MPC problem (possibly LTV, Linear Time Varying):

$$
\begin{aligned}
V^*(p) = \min_{z} \quad & \ell_N(x_N) + \sum_{k=0}^{N-1} \ell_k(x_k, u_k) \\
\text{s.t.} \quad & x_0 = p \\
& x_{k+1} = A_k x_k + B_k u_k + f_k \\
& F_k x_k + G_k u_k \leq c_k \\
& \qquad k = 0, \ldots, N-1 \\
& F_N x_N \leq c_N
\end{aligned}
$$

quadratic costs

linear (affine) prediction model

linear inequality constraints

- QP problem:

$$
\begin{aligned}
V^\star(p) \triangleq \min_{z} \quad & \frac{1}{2} z' M z + (Cp + g)' z + \frac{1}{2} p' Y p \\
\text{s.t.} \quad & Gz \leq Ep + b
\end{aligned}
$$

$z =$ optimization vector

$p = x(t)$ parameter vector

Dual QP problem

$$\begin{aligned} V^\star(p) \triangleq \min_z \quad & \frac{1}{2}z'Mz + (Cp+g)'z + \frac{1}{2}p'Yp \\ \text{s.t.} \quad & Gz \leq Ep + b \end{aligned}$$

original (primal) QP

$$\begin{aligned} H &= GM^{-1}G' \\ D &= GM^{-1}C + E \\ d &= GM^{-1}g + b \end{aligned}$$

$$\begin{aligned} \Phi^\star(p) \triangleq \min_y \quad & \frac{1}{2}y'Hy + (Dp+d)'y + d_p \\ \text{s.t.} \quad & y \geq 0 \end{aligned}$$

dual QP

- Dual QP has simpler constraint set (=orthant) to project onto !

$$\text{proj}_{\{y \geq 0\}}\ y = \max\{y, 0\}$$

- **Key idea:** apply standard **fast gradient projection** algorithm to solve dual QP problem

(Nesterov, 1983)

Fast gradient projection algorithm

- **Off-line** operations: = linear algebra computations on QP matrices

$$M_G \triangleq M^{-1}G' \qquad G_L \triangleq \frac{1}{L}G$$
$$M_C \triangleq M^{-1}C \qquad E_L \triangleq \frac{1}{L}E$$
$$m_g \triangleq M^{-1}g \qquad b_L \triangleq \frac{1}{L}b$$

L = max eigenvalue of H
or
$$L = \sqrt{\sum_{i,j=1}^{m} |H_{i,j}|^2}$$ Frobenius norm

- **On-line** operations: initialization given current p=x(t)

$$g_P \triangleq M_C p + m_g, \quad p_D \triangleq -E_L p - b_L$$
$$y_0 = y_{-1} = 0, \quad z_{-1} = 0, \quad \theta_0 = \theta_{-1} = 1$$

- **On-line** operations: iterations

$$\theta_{\nu+1} = \frac{\sqrt{\theta_\nu^4+4\theta_\nu^2}-\theta_\nu^2}{2}$$
$$\beta_\nu = \theta_\nu(\theta_{\nu-1}^{-1} - 1)$$

$$\begin{cases} w_\nu = y_\nu + \beta_\nu(y_\nu - y_{\nu-1}) \\ \hat{z}_\nu = -M_G w_\nu - g_P \\ z_\nu = (1-\theta_\nu)z_{\nu-1} + \theta_\nu \hat{z}_\nu \\ y_{\nu+1} = \max\{w_\nu + G_L\hat{z}_\nu + p_D, 0\} \end{cases}$$

$\theta, \beta \in \mathbb{R}$, ν=iteration counter

Complexity (code, operations)

- Main on-line operations involve only **simple linear algebra**

$$\begin{cases} w_\nu &= y_\nu + \beta_\nu (y_\nu - y_{\nu-1}) \\ \hat{z}_\nu &= -M_G w_\nu - g_P \\ z_\nu &= (1-\theta_\nu) z_{\nu-1} + \theta_\nu \hat{z}_\nu \\ y_{\nu+1} &= \max\{w_\nu + G_L \hat{z}_\nu + p_D, 0\} \end{cases}$$

matrix-vector products

projection operation = simple comparison !

- Few lines of **`(Embedded) MATLAB`** or **`Python/Numpy`** code !

- Operations can be easily **parallelized** (for example on **`GPU`**)

Complexity (code, operations)

$$\begin{cases} w_\nu &= y_\nu + \beta_\nu(y_\nu - y_{\nu-1}) \\ \hat{z}_\nu &= -M_G w_\nu - g_P \\ z_\nu &= (1-\theta_\nu) z_{\nu-1} + \theta_\nu \hat{z}_\nu \\ y_{\nu+1} &= \max\{w_\nu + G_L \hat{z}_\nu + p_D, 0\} \end{cases}$$

- Each iteration has complexity $\mathrm{O}(N^2)$ $[N =$ prediction horizon$]$ (matrix-vector product)

- Using Riccati-like iterations, complexity gets down to $O(N)$

(Patrinos, Bemporad, CDC 2012)

- For **LTV problems,** all matrices must be computed on-line

Termination criteria

- We are interested in quality of **primal solution**

$$\begin{aligned} V(z,p) - V^\star(p) &\le \varepsilon_V \\ \max_{i \in \mathbb{N}_{[1,m]}} G_i z - E_i p - b_i &\le \varepsilon_g \end{aligned}$$

$$\begin{aligned} \varepsilon_V &\ge 0 \quad \text{optimality tolerance} \\ \varepsilon_g &\ge 0 \quad \text{feasibility tolerance} \end{aligned}$$

$$V(z,p) = \frac{1}{2} z' M z + (Cp + g)' z + \frac{1}{2} p' Y p$$

primal cost function

- V^* is unknown, so we use the duality gap as a **stopping criterion**:

$$\begin{aligned} V(z_\nu, p) - \Phi(y_{\nu+1}, p) &\le \varepsilon_V \\ \max_{i \in \mathbb{N}_{[1,m]}} G_i z_\nu - E_i p - b_i &\le \varepsilon_g \end{aligned}$$

$$\Phi(y,p) = \frac{1}{2} y' H y + (Dp + d)' y + d_p$$

dual cost function

Bounds on the number of iterations

(Patrinos, Bemporad, 2012)

- Feasibility:

$$G z_\nu - Ep - b \leq \varepsilon_g \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \quad \Longrightarrow \quad \forall \nu \geq \left\lceil \sqrt{\frac{8L\|y^*(p)\|}{\varepsilon_g}} \right\rceil - 2$$

- Optimality:

$\forall y^*(p)$ solving dual problem

$$\begin{gathered} V(z_\nu, p) - \Phi(y_{\nu+1}) \leq \varepsilon_V \\ \Downarrow \\ V(z_\nu, p) - V^\star(p) \leq \varepsilon_V \end{gathered} \quad \Longrightarrow \quad \forall \nu \geq \left\lceil \sqrt{\frac{2L}{\varepsilon_V}} \|y^*(p)\| \right\rceil - 2$$

(and also $V(z_\nu, p) - V^\star(p) \geq -\varepsilon_V$, see proof)

- A bound on $\|z_\nu - z^*(p)\|$ exists too

The bounds depend on optimal dual solution $y^*(p)$ ...

Bounds on the number of iterations

- A global bound on $\|y^*(p)\|$ for a given set $\mathcal{P}$ of parameters can be computed **off-line** by solving a **MILP** or a **LPCC** (Linear Program with linear Complementarity Constraints)

(Patrinos, Bemporad, 2012)

- When no constraints are active, solution is found in **one step:** $y_0 = y^* = 0, \;\; z^* = -M^{-1}(Cp+g)$ is the optimal solution

- More generally: QP optimality conditions can be stated as

$$y^* = \max\{y^* - \frac{1}{L}(Hy^* + Dp + d), 0\}$$

$$z^* = -M^{-1}(G'y^* + Cp + g)$$

Solving QP = finding a solution of the system of PWA equations

GPAD vs other fast gradient projection methods

- Nesterov's method applied to primal problem, only input-constrained MPC with simple input constraints (Richter, Jones, Morari, CDC 2009)

- Relax state equations, solve the dual. Only simple constraints and terminal sets. (Richter, Morari, Jones, CDC 2011, IEEE TAC 2012)

- Richter's algorithms must be run for a constant number of iterations (= theoretical bound), no termination criteria based on primal solution

- GPAD relaxes linear inequality constraints, not state equations. Can handle **arbitrary polyhedral constraints on inputs/states** (and polyhedral terminal sets). Practical **termination criteria based on primal solution** are provided. Different **theoretical bounds** are proved.

GPAD vs Interior Point (IP) methods

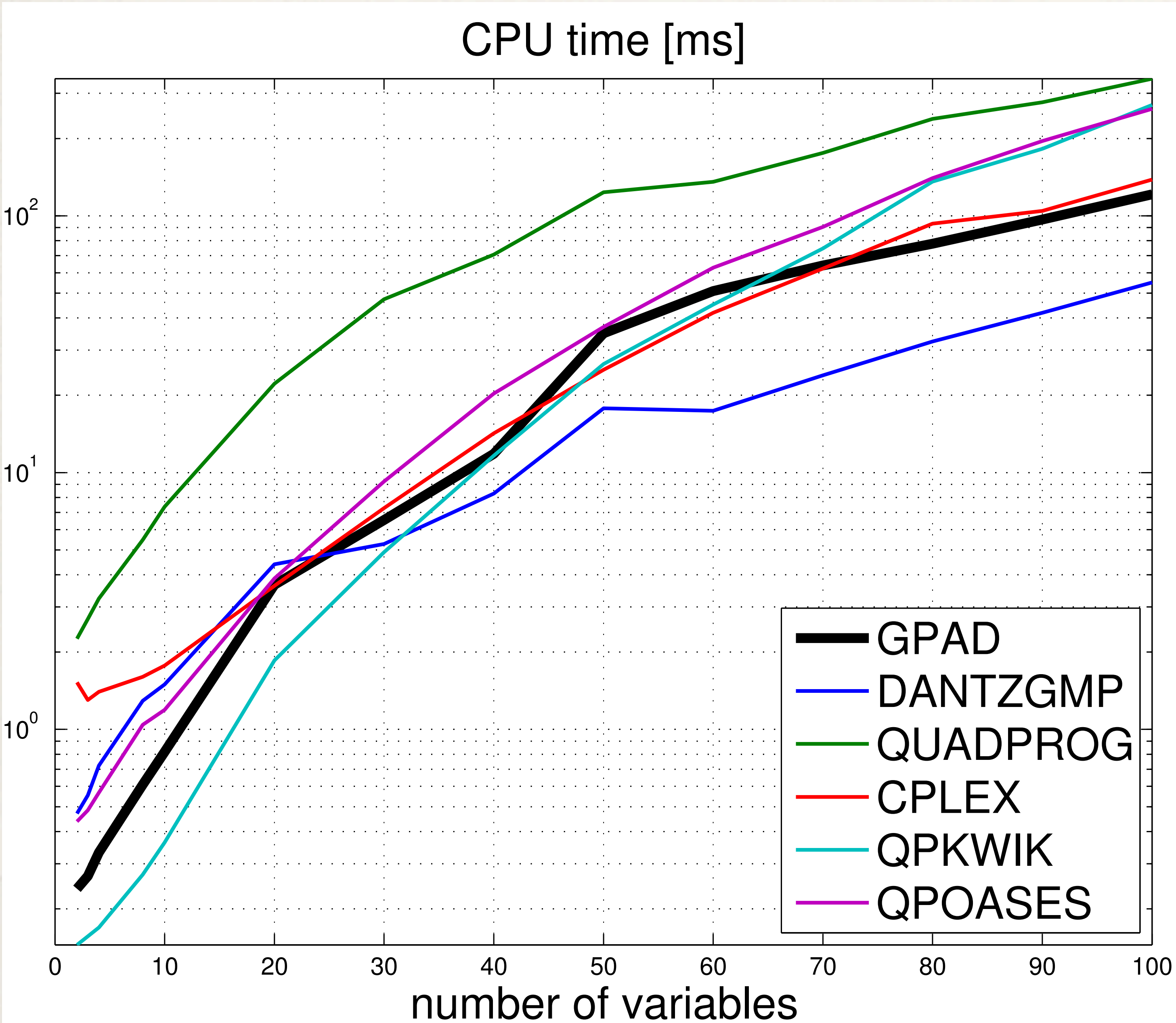
- GPAD iterations only require a **matrix-vector product,** IP requires solving a linear system

- GPAD can reach a solution of **moderate accuracy quite fast**, faster than a simple gradient method (both in practice and in theory). This is often enough for MPC purposes

- GPAD has much **tighter theoretical complexity** bounds than IP

- GPAD is **much simpler to code** !

Numerical results: speed



Embedded MATLAB code

```matlab
while keepgoing && (nu<maxiter),
    beta=th*(inv(th0)-1);
    w=y+beta*(y-y0);
    zhat=-(MG*w+gP);
    z=(1-th)*z+th*zhat;

    th0=th; % Update th0
    th2=th^2;
    th=(sqrt(th2^2+4*th2)-th2)/2;
    y0=y;
    y=max(w+GL*zhat+pD,0);

    gaphat=.5*(y'*H*y+zhat'*M*zhat)+...
        dp+c'*zhat+Dpd'*y;
    violhat=max(G*zhat-b);
    if gaphat<=epsV && violhat<=epsG,
        keepgoing=0;
    end
    nu=nu+1;
end
```

Average CPU time on random QP problems with n variables and $2n$ constraints

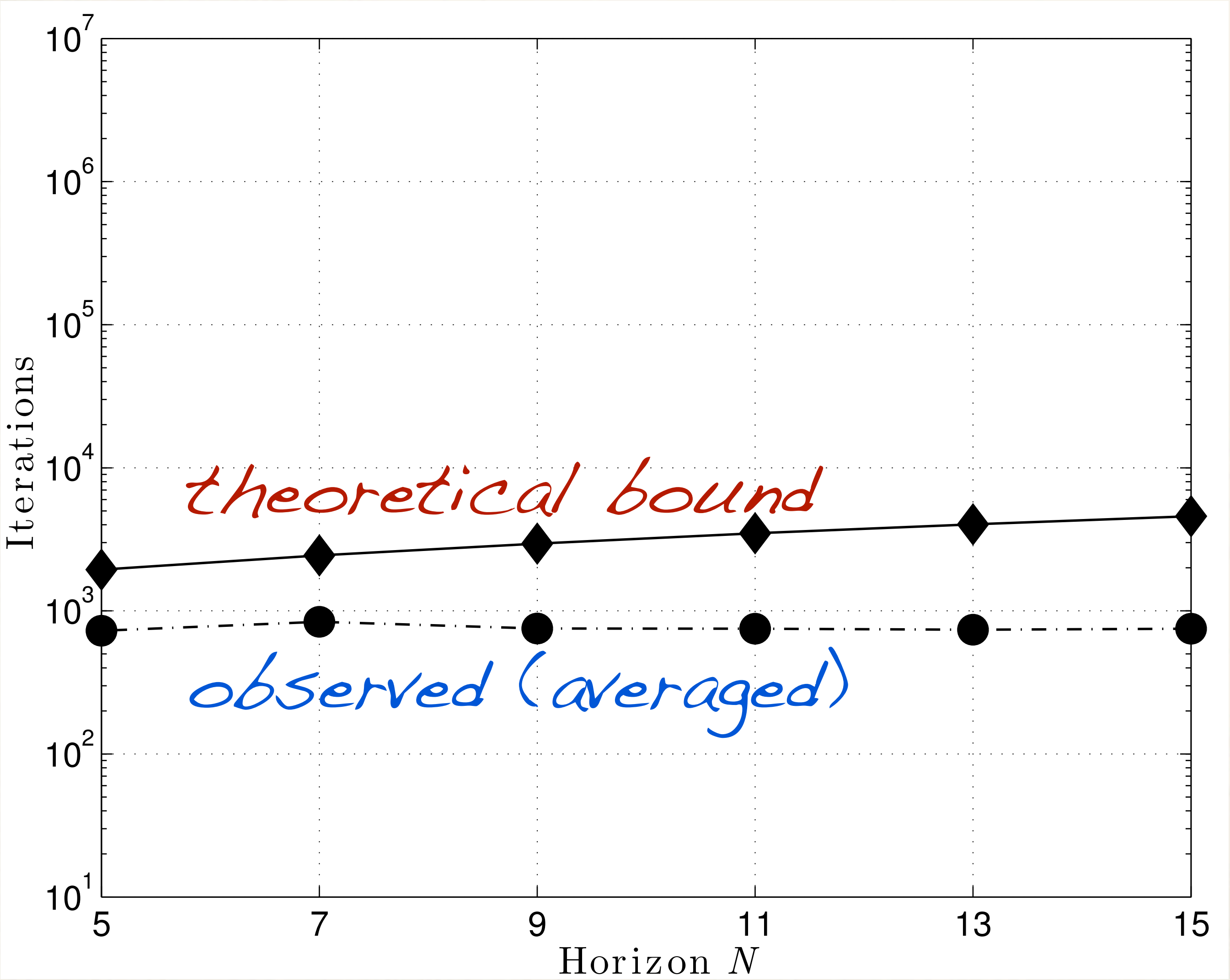
[standard algorithm with complexity $\mathrm{O}(n^2)$]

Numerical results: bounds

Ball & plate example, 2 states, 1 input, bound constraints on states and inputs

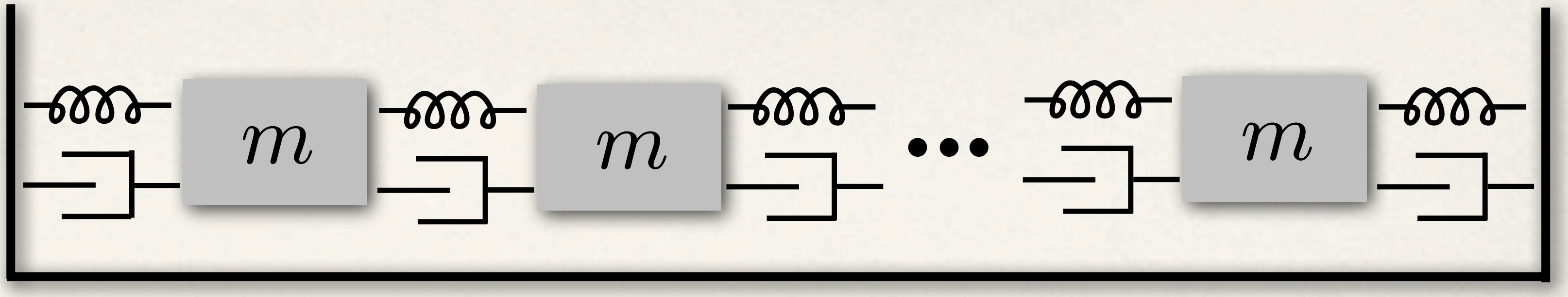
(Richter, Jones, Morari, 2011)

(Waldvogel,2010)



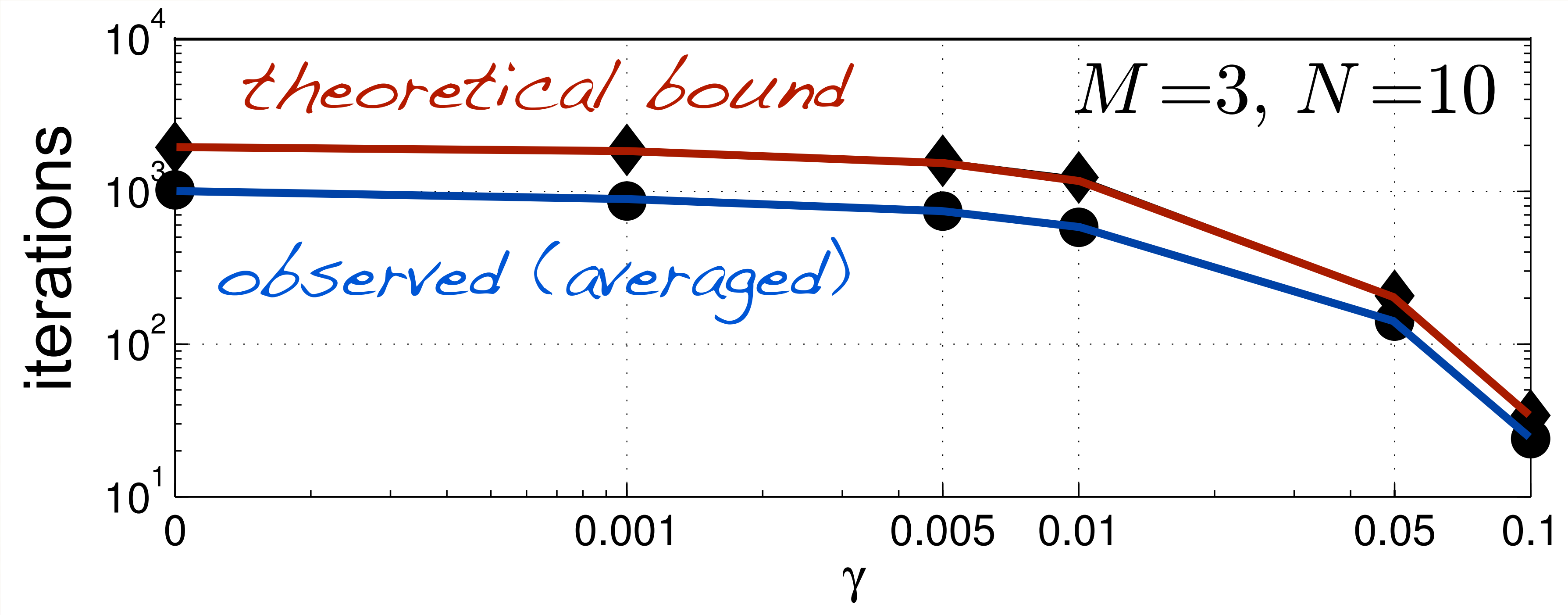
Numerical results: more complex example

Linear system:



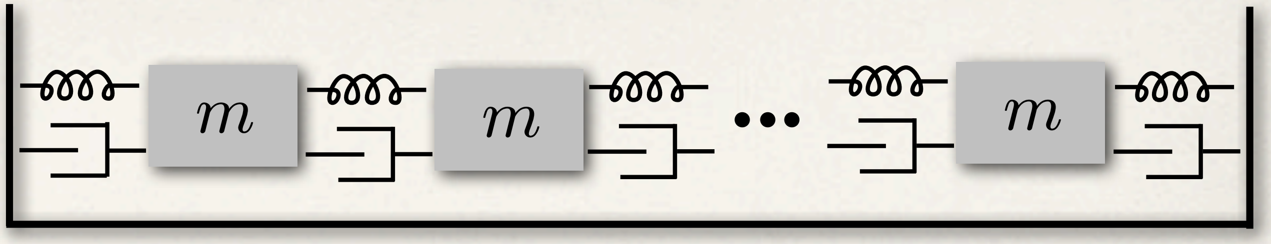
(Wang, Boyd, 2008)

- $2M$ states, M-1 inputs
- state and input constraints
- terminal set



γ = shrinking of parameter set, $\mathcal{P}_\gamma \triangleq \mathrm{dom}\{z \in \mathcal{Z}(p) | g(z) + \gamma \leq 0\}$

Numerical results: more complex example



CPU time [ms]

M	N	GPAD		Gurobi 5.0	
		avg	max	avg	max
2	10	0.40	1.09	4.48	
2	30	0.76	3.22	5.64	9.77
4	10	1.41	2.63	5.61	7.49
6	30	8.35	15.52	15.22	23.57
8	20	8.65	14.95	16.42	18.36
15	10	11.00	16.62	21.52	22.95
20	20	39.01	82.32	82.71	134.05
30	30	128.1	218.13	202.35	465.80

number of iterations

M	N	GPAD		Gurobi 5.0	
		avg	max	avg	max
2	10	19.90	60	5.42	7
2	30	20.40	100	5.50	7
4	10	41.80	80	6.05	8
6	30	52.20	100	6.57	8
8	20	52.20	90	6.51	8
15	10	54.00	80	6.56	7
20	20	61.60	140	6.85	8
30	30	62.00	100	6.94	8

QP **interior point** solver of `Gurobi 5.0`

tolerances: $\epsilon_V = \epsilon_g = 10^{-3}$

results averaged on 100 random states p

Conclusions

- GPAD solver for **QP tailored to embedded MPC** applications
  - Very **simple to code**
  - **Tight bounds** on estimated real-time execution
  - **Quality control of solution** (primal optimality & feasibility criteria)
  - Although not conceived for speed, still **reasonably fast**

- Current ongoing work
  - Robustness to computation errors (=fixed-point implementation)
  - Closed-loop stability of MPC under lower-quality solutions
  - Combined GPAD and Newton's method (for faster convergence)
  - Applications to aero industry (with ESA / Astrium / A3R)

Conclusions

Research on QP started ~60 years ago …

ON MINIMIZING A CONVEX FUNCTION SUBJECT TO LINEAR INEQUALITIES

By E. M. L. BEALE

Admiralty Research Laboratory, Teddington, Middlesex

SUMMARY

THE minimization of a convex function of variables subject to linear inequalities is discussed briefly in general terms. Dantzig's Simplex Method is extended to yield finite algorithms for minimizing either a convex quadratic function or the sum of the t largest of a set of linear functions, and the solution of a generalization of the latter problem is indicated. In the last two sections a form of linear programming with random variables as coefficients is described, and shown to involve the minimization of a convex function.

(Beale, 1955)

… still research on new **QP solvers for MPC** can have most impact !