NSIPS: Nonlinear Semi-Infinite Programming Solver

A. Ismael F. Vaz
Edite M.G.P. Fernandes
Production and Systems Department
Engineering School
Minho University
Braga - Portugal
{aivaz,emgpf}@dps.uminho.pt

M. Paula S.F. Gomes
Mechanical Engineering Department
Imperial College of Science, Technology and Medicine
London SW7 2BX - UK
p.gomes@ic.ac.uk

18-20 June, 2004

Talk partially financed by the Portuguese Calouste Gulbenkian Foundation

Contents

- Semi-Infinite Programming (SIP)
- Nonlinear SIP Solver - (NSIPS)
  - ⋆ Discretization method
  - ⋆ Sequential quadratic programming method
  - ⋆ Constraint transcription methods
    - ∗ Interior point method
    - ∗ Penalty (multiplier) method
- Numerical results / Conclusions

Semi-Infinite Programming

$$\begin{aligned} &\min_{x \in R^n} \ f(x) \\ s.t. \ \ & g_i(x,t) \leq 0, \ i = 1, ..., m \\ & h_i(x) \leq 0, \ i = 1, ..., o \\ & h_i(x) = 0, \ i = o+1, ..., q \\ & \forall t \in T \end{aligned} \tag{1}$$

where $f(x)$ is the objective function, $h_i(x)$ are the finite constraint functions, $g_i(x,t)$ are the infinite constraint functions and $T \subset R^p$ is, usually, a cartesian product of intervals $([\alpha_1, \beta_1] \times [\alpha_2, \beta_2] \times \cdots \times [\alpha_p, \beta_p])$.

Solver

NSIPS (Nonlinear Semi-Infinite Programming Solver)

Solver

NSIPS (Nonlinear Semi-Infinite Programming Solver)

Version 2.1 publicly available in the internet `http://www.norg.uminho.pt/aivaz/`, implementing the four methods in a total of seven algorithms.

Solver

NSIPS (Nonlinear Semi-Infinite Programming Solver)

Version 2.1 publicly available in the internet `http://www.norg.uminho.pt/aivaz/`, implementing the four methods in a total of seven algorithms.

NSIPS receives the problem in the (SIP)AMPL format.

Solver

NSIPS (Nonlinear Semi-Infinite Programming Solver)

Version 2.1 publicly available in the internet `http://www.norg.uminho.pt/aivaz/`, implementing the four methods in a total of seven algorithms.

NSIPS receives the problem in the (SIP)AMPL format.

The method is select by the *nsips_options*, and there are many other options for each method.

Solver

NSIPS (Nonlinear Semi-Infinite Programming Solver)

Version 2.1 publicly available in the internet `http://www.norg.uminho.pt/aivaz/`, implementing the four methods in a total of seven algorithms.

NSIPS receives the problem in the (SIP)AMPL format.

The method is select by the *nsips_options*, and there are many other options for each method.

The NPSOL software is used to solve the finite subproblems (discretization and SQP methods).

Solver

NSIPS (Nonlinear Semi-Infinite Programming Solver)

Version 2.1 publicly available in the internet `http://www.norg.uminho.pt/aivaz/`, implementing the four methods in a total of seven algorithms.

NSIPS receives the problem in the (SIP)AMPL format.

The method is select by the *nsips_options*, and there are many other options for each method.

The NPSOL software is used to solve the finite subproblems (discretization and SQP methods).

NSIPS is available in the NEOS server (`www-neos.mcs.anl.gov`).

Discretization method - Three versions

A sequence of finite problems are solved. The finite problems are obtained from the SIP problem where the infinite constraints are evaluated at a finite set of points $\tilde{T}[h^k] \subseteq T[h^k]$, where $T[h^k] \subseteq T$ is a uniform grid of points with space h^k.

Versions adapted for nonlinear SIP and implemented:

- Hettich (1986, 1990)
- Reemtsen (1991)
- Hettich with pseudo-number generation.

Discretization method

- STEP 0: Define $T[h^0]$. Let $\widetilde{T}[h^0] = T[h^0]$. Solve the NLP($\widetilde{T}[h^0]$) and let x_0 be the solution found.

- STEP k: If x_{k-1} is not feasible for all the points in the set $T[h^{k-1}]$
  - ⋆ THEN: Insert all the infeasible points in the set $\widetilde{T}[h^{k-1}]$. Solve the NLP($\widetilde{T}[h^{k-1}]$) and let x_{k-1} be the solution found. Continue with step k.
  - ⋆ ELSE: If the maximum number of refinements is reached then stop. Else build the set $\widetilde{T}[h^k]$ from $T[h^k]$ and $\widetilde{T}[h^{k-1}]$. Solve the NLP($\widetilde{T}[h^k]$) and let x_k be the solution found. Go to step $k+1$.

Reduced problem

Problem with no finite constraints and only one infinite variable.

$$\begin{aligned} &\min_{x\in R^n} \ f(x) \\ s.t.\ \ &g_i(x,t) \leq 0,\ i=1,...,m \\ &\forall t \in T \equiv [a,b] \end{aligned} \tag{2}$$

Sequential Quadratic Programming

Considering the reduced problem (2), the sequential quadratic programming is based on the quadratic semi-infinite programming (QSI)

$$
\begin{gathered}
\min_{d \in R^n} \; f_Q(d) = \frac{1}{2} d^T H_k d + d^T \nabla f(x_k) \\
s.t. \;\; d^T \nabla_x g_i(x_k, t) + g_i(x_k, t) \leq 0, \\
i = 1, \ldots, m, \; \forall t \in [a, b] \;\; ,
\end{gathered}
$$

where H_k is an approximation to $\nabla^2_{xx} \mathcal{L}(x_k, v)$.

SQP

The solution of the QSI problem is d_k and

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 1, 2, \ldots$$

$\{x_k\} \to x^*$, solution to the initial SIP problem.

The Lagrangian of the QSI problem is

$$\mathcal{L}_Q(d, v) = \frac{1}{2} d^T H_k d + d^T \nabla f(x_k) \\ + \sum_{j=1}^{m} \int_a^b \left(d^T \nabla_x g_j(x_k, t) + g_j(x_k, t) \right) dv_j(t)$$

Solving the QSI

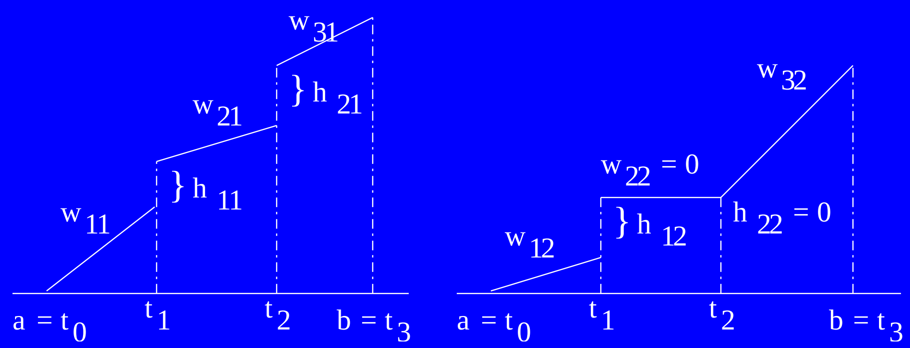
The dual problem $\min_{v\in\mathcal{V}^*} \mathcal{L}_Q^*(v) \equiv -\mathcal{L}_Q(d(v),v)$ is solved by a linear parametrization of the dual variables.

$$v_j(t) = \begin{cases} w_{1j}(t-a), & \text{for } t \in [a,t_1); \\ a_{ij} + w_{i+1j}(t-t_i), & \text{for } t \in [t_i,t_{i+1}), \\ & i = 1,2,\ldots,l-1; \\ a_{lj} + w_{l+1j}(t-t_l), & \text{for } t \in [t_l,b]\,; \end{cases}$$

$j = 1,\ldots,m$, where

$$a_{ij} = \sum_{p=1}^{i} h_{pj} + \sum_{p=1}^{i} w_{pj}(t_p - t_{p-1}),\ i = 1,\ldots,l$$

Example with $m = 2$, $l = 2$



w are the linear segments slope, h are the jumps and t are the discontinuity points.

Merit function

$$\phi(x, \mu) = f(x) + \frac{1}{2}\mu \sum_{i=1}^{m} \int_{a}^{b} [g_i(x, t)]_+^2 dt$$

where $[z]_+ = \max\{0, z\}$.

A strategy for computing the penalty parameter.

Numerical integrals computation - Numerical adaptative formulae (Gaussian or trapezoid).

SQP - Dual method

1. Given x_0. Let $k = 0$ and $H_0 = I$.

2. Compute H_k using a BFGS quasi-Newton updating formula.

3. Solve the QSI problem to obtain the search direction d_k.

4. If $d_k = \mathbf{0}$ then stop.

5. Find α_k such that $x_{k+1} = x_k + \alpha_k d_k$ sufficiently decreases the merit function.

6. If there is not a major difference between x_{k+1} and x_k then stop with x_{k+1} as an approximated solution. Otherwise do $k = k + 1$ and go to step 2.

Constraint transcription

Considering the reduced problem (2), the infinite constraints $g_i(x,t) \leq 0$, $\forall t \in T$, are transformed into $\int_T [g_i(x,t)]_+ dt = 0$ where $[z]_+ = \max\{0, z\}$.

The SIP is then transformed into

$$\min_{x \in R^n} \; f(x)$$

$$s.t. \;\; G_i(x) \equiv \int_T [g_i(x,t)]_+ dt = 0$$

$$i = 1, \ldots, m$$

Constraint functions not differentiable.

Approximate problem

$$\min_{x \in R^n} \; f(x)$$

$$s.t. \;\; G_{i,\epsilon}(x) \equiv \int_T g_{i,\epsilon}(x,t)dt = 0$$

$$i = 1, \ldots, m$$

with $\epsilon \longrightarrow 0$ and

$$g_{i,\epsilon}(x,t) = \begin{cases} 0, & \text{if } \; g_i(x,t) < -\epsilon; \\ \frac{(g_i(x,t)+\epsilon)^2}{4\epsilon}, & \text{if } \; -\epsilon \leq g_i(x,t) \leq \epsilon; \\ g_i(x,t), & \text{if } \; g_i(x,t) > \epsilon, \end{cases}$$

Once differentiable constraint functions.

Penalty method

A sequence of subproblems is solved, parameterized by μ

$$\min_{x \in R^n} \phi_S(x, \mu)$$

for a sequence of increasing $\mu > 0$ values.

Simple penalty functions

$$\phi_S^1(x,\mu) = f(x) + \mu \sum_{i=1}^{m} \int_T g_{i,\epsilon}(x,t)dt$$

Simple penalty functions

$$\phi_S^1(x,\mu) = f(x) + \mu \sum_{i=1}^{m} \int_T g_{i,\epsilon}(x,t)dt$$

$$\phi_S^2(x,\mu) = f(x) + \frac{\mu}{2} \sum_{i=1}^{m} \int_T g_{i,\epsilon}(x,t)^2 dt$$

Simple penalty functions

$$\phi_S^1(x,\mu) = f(x) + \mu \sum_{i=1}^{m} \int_T g_{i,\epsilon}(x,t)dt$$

$$\phi_S^2(x,\mu) = f(x) + \frac{\mu}{2} \sum_{i=1}^{m} \int_T g_{i,\epsilon}(x,t)^2 dt$$

and

$$\phi_S^3(x,\mu) = f(x) + \mu \sum_{i=1}^{m} \int_T \left(e^{g_{i,\epsilon}(x,t)} - 1 \right) dt$$

Relaxed problem to satisfy LICQ

$$\min_{x \in R^n} \ f(x)$$

$$s.t. \ \ G_{i,\epsilon}(x) \leq \tau$$

$$i = 1, \ldots, m$$

$$\tau > 0 \quad (\tau(\epsilon) \to 0)$$

Multiplier method

A sequence of subproblems is solved

$$\min_{x \in R^n} \phi_{AL}(x, \lambda, \mu)$$

where ϕ_{AL} is the augmented Lagrangian penalty function

Multiplier method

A sequence of subproblems is solved

$$\min_{x \in R^n} \phi_{AL}(x, \lambda, \mu)$$

where ϕ_{AL} is the augmented Lagrangian penalty function

$$\phi_{AL}(x,\lambda,\mu) = f(x) + \sum_{i=1}^{m} \lambda_i \left(\int_T g_{i,\epsilon}(x,t)dt - \tau \right) + \frac{\mu}{2} \sum_{i=1}^{m} \left(\int_T g_{i,\epsilon}(x,t)dt \right)^2$$

where $\lambda = (\lambda_1, \ldots, \lambda_m)^T$ is the Lagrange multipliers vector.

Lagrange multipliers update

Since the optimum Lagrange multipliers are unknown before computing the solution, an updating formula for the Lagrange multipliers is used.

$$\lambda_i^{k+1} = \lambda_i^k + \mu^k \int_T g_{i,\epsilon}(x^k, t)dt, \quad i = 1, \dots, m.$$

Multiplier method

A sequence of subproblems is solved

$$\min_{x \in R^n} \phi_E(x, \lambda, \mu)$$

where ϕ_E is the exponential penalty function

Multiplier method

A sequence of subproblems is solved

$$\min_{x \in R^n} \phi_E(x, \lambda, \mu)$$

where ϕ_E is the exponential penalty function

$$\begin{aligned}\phi_E(x, \lambda, \mu) = & f(x) \\ & + \frac{1}{\mu} \sum_{i=1}^{m} \lambda_i \left(e^{\mu \left(\int_T g_{i,\epsilon}(x,t) dt - \tau \right)} - 1 \right)\end{aligned}$$

Multiplier method

A sequence of subproblems is solved

$$\min_{x \in R^n} \phi_E(x, \lambda, \mu)$$

where ϕ_E is the exponential penalty function

$$\begin{aligned} \phi_E(x, \lambda, \mu) =& f(x) \\ & + \frac{1}{\mu} \sum_{i=1}^{m} \lambda_i \left(e^{\mu \left(\int_T g_{i,\epsilon}(x,t) dt - \tau \right)} - 1 \right) \end{aligned}$$

An updating formula for the Lagrange multipliers is used

$$\lambda_i^{k+1} = \lambda_i^k e^{\mu^k \left(\int_T g_{i,\epsilon}(x^k,t) dt - \tau \right)}, \quad i = 1, \ldots, m$$

Multiplier penalty framework

1. Given an initial guess for x and λ, and parameters μ, ϵ and $\tau(\epsilon)$.

2. Exterior iteration. The initial guess for the interior iterations is the last approximation computed.

3. Interior iterations. For μ and λ, solve the unconstrained problem

   $$\min_{x \in R^n} \phi(x, \lambda, \mu)$$

   through a BFGS quasi-Newton technique and a line search with an Armijo like rule that significantly reduces the penalty function.

   Solution: $x^*(\mu)$.

4. If the computed approximation is infeasible ($\int_T g_{i,\epsilon}(x^*(\mu),t)dt - \tau > 0,\ i = 1,...,m$) then update the penalty parameter μ, the multipliers vector λ and proceed with another exterior iteration.

5. Otherwise, if there is a significant evolution from the last two approximations computed for different differentiable parameters (ϵ e $\tau(\epsilon)$) then update the differentiability parameter and proceed with another exterior iteration.

6. Stop with the last computed approximation being an approximation to the SIP solution ($x^* \leftarrow x^*(\mu)$).

Primal-dual interior point method

From the relaxed problem, the barrier problem is formed by placing the slack variables in the barrier term

$$\min_{x \in R^n, s \in R^m} f(x) - \mu \sum_{i=1}^{m} \log(s_i + \tau)$$

$$s.t. \quad \int_T g_{i,\epsilon}(x,t)dt + s_i = 0, \quad i = 1, \ldots, m$$

with $g_{i,\epsilon}(x,t) = \frac{g_i(x,t)+\sqrt{g_i(x,t)^2+\epsilon^2}}{2}$ and $\epsilon \to 0$ $(\epsilon > 0)$.

The barrier problem is solved for a sequence of $\mu(\to 0)$ values.

By applying the Newton method to the first order KKT system:

Newton system

$$\begin{pmatrix} H & 0 & J \\ 0 & \Lambda & S \\ -J^T & -I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} \sigma \\ \gamma \\ \rho \end{pmatrix}$$

with

$$H = \nabla^2 f - \sum_{i=1}^{m} \lambda_i \int_T \nabla_{xx}^2 g_{i,\epsilon}(x,t) dt$$

Newton system

$$\begin{pmatrix} H & 0 & J \\ 0 & \Lambda & S \\ -J^T & -I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} \sigma \\ \gamma \\ \rho \end{pmatrix}$$

with

$$H = \nabla^2 f - \sum_{i=1}^{m} \lambda_i \int_T \nabla^2_{xx} g_{i,\epsilon}(x,t)dt$$

$$J = \left(\int_T \nabla_x g_{1,\epsilon}(x,t)dt, \ldots, \int_T \nabla_x g_{m,\epsilon}(x,t)dt \right)$$

Newton system

$$\begin{pmatrix} H & 0 & J \\ 0 & \Lambda & S \\ -J^T & -I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} \sigma \\ \gamma \\ \rho \end{pmatrix}$$

with

$$H = \nabla^2 f - \sum_{i=1}^{m} \lambda_i \int_T \nabla^2_{xx} g_{i,\epsilon}(x,t)dt$$

$$J = \left(\int_T \nabla_x g_{1,\epsilon}(x,t)dt, \ldots, \int_T \nabla_x g_{m,\epsilon}(x,t)dt \right)$$

$$S = diag(s_i + \tau),\ \Lambda = diag(\lambda_i), \quad i = 1, \ldots, m$$

Newton system

$$\begin{pmatrix} H & 0 & J \\ 0 & \Lambda & S \\ -J^T & -I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} \sigma \\ \gamma \\ \rho \end{pmatrix}$$

with

$$I = \text{Identity matrix}$$

Newton system

$$\begin{pmatrix} H & 0 & J \\ 0 & \Lambda & S \\ -J^T & -I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} \sigma \\ \gamma \\ \rho \end{pmatrix}$$

with

$$I = \text{Identity matrix}$$

$$\sigma = -\nabla f - J\lambda \quad \text{(Dual infeasibility)}$$

Newton system

$$\begin{pmatrix} H & 0 & J \\ 0 & \Lambda & S \\ -J^T & -I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} \sigma \\ \gamma \\ \rho \end{pmatrix}$$

with

$$I = \text{Identity matrix}$$

$$\sigma = -\nabla f - J\lambda \quad \text{(Dual infeasibility)}$$

$$\gamma = \mu e - S\Lambda e$$

Newton system

$$\begin{pmatrix} H & 0 & J \\ 0 & \Lambda & S \\ -J^T & -I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} \sigma \\ \gamma \\ \rho \end{pmatrix}$$

with

$$I = \text{Identity matrix}$$

$$\sigma = -\nabla f - J\lambda \quad \text{(Dual infeasibility)}$$

$$\gamma = \mu e - S\Lambda e$$

$$\rho = \bar{g} + s \quad \text{(Primal infeasibility)}$$

Newton system

$$\begin{pmatrix} H & 0 & J \\ 0 & \Lambda & S \\ -J^T & -I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} \sigma \\ \gamma \\ \rho \end{pmatrix}$$

with

$$I = \text{Identity matrix}$$

$$\sigma = -\nabla f - J\lambda \quad \text{(Dual infeasibility)}$$

$$\gamma = \mu e - S\Lambda e$$

$$\rho = \bar{g} + s \quad \text{(Primal infeasibility)}$$

$$\bar{g} = (G_{1,\epsilon}, \ldots, G_{m,\epsilon})^T$$

Newton system

$$\begin{pmatrix} H & 0 & J \\ 0 & \Lambda & S \\ -J^T & -I & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} \sigma \\ \gamma \\ \rho \end{pmatrix}$$

$(\Delta x, \Delta s, \Delta \lambda)$ is the Newton direction and

$$x_{k+1} = x_k + \alpha_k \Delta x_k$$

$$s_{k+1} = s_k + \alpha_k \Delta s_k$$

$$\lambda_{k+1} = \lambda_k + \alpha_k \Delta \lambda_k$$

Implemented merit functions

Choosing α to obtain feasibility and convergence to the minimum.

$$\phi(x, s; \mu, \beta) = f(x) - \mu \sum_{i=1}^{m} \log(s_i + \tau) + \frac{\beta}{2} \rho^T \rho$$

Implemented merit functions

Choosing α to obtain feasibility and convergence to the minimum.

$$\phi(x, s; \mu, \beta) = f(x) - \mu \sum_{i=1}^{m} \log(s_i + \tau) + \frac{\beta}{2} \rho^T \rho$$

$$\begin{aligned} \mathcal{L}_A(x, s, \lambda; \mu, \beta) = & f(x) - \mu \sum_{i=1}^{m} \log(s_i + \tau) + \lambda^T \rho \\ & + \frac{\beta}{2} \rho^T \rho \end{aligned}$$

Quasi-Newton strategy with an approximation to the Hessian of the Lagrangian.

Primal-dual interior point algorithm

1. Given x_0, ϵ, τ, θ, δ_μ and δ_f.

2. Compute $s_{i,0}$ and $\lambda_{i,0}$, $i = 1, \ldots, m$. Let $k = 0$.

3. Let $y_{eps} = x_k$ the last y computed for a given ϵ.

4. Compute or update μ_k.

5. Stopping criteria. If the stopping criteria is verified then if there is a significant difference between y_{eps} and x_k reduce ϵ, τ, update the slack variables and go to step 3; Otherwise stop.

6. Update B_k by a BFGS formula. If $k = 0$ then $B_k =$Identity matrix.

7. Solve the KKT system to obtain the search direction $(\Delta x_k, \Delta s_k, \Delta \lambda_k)$.

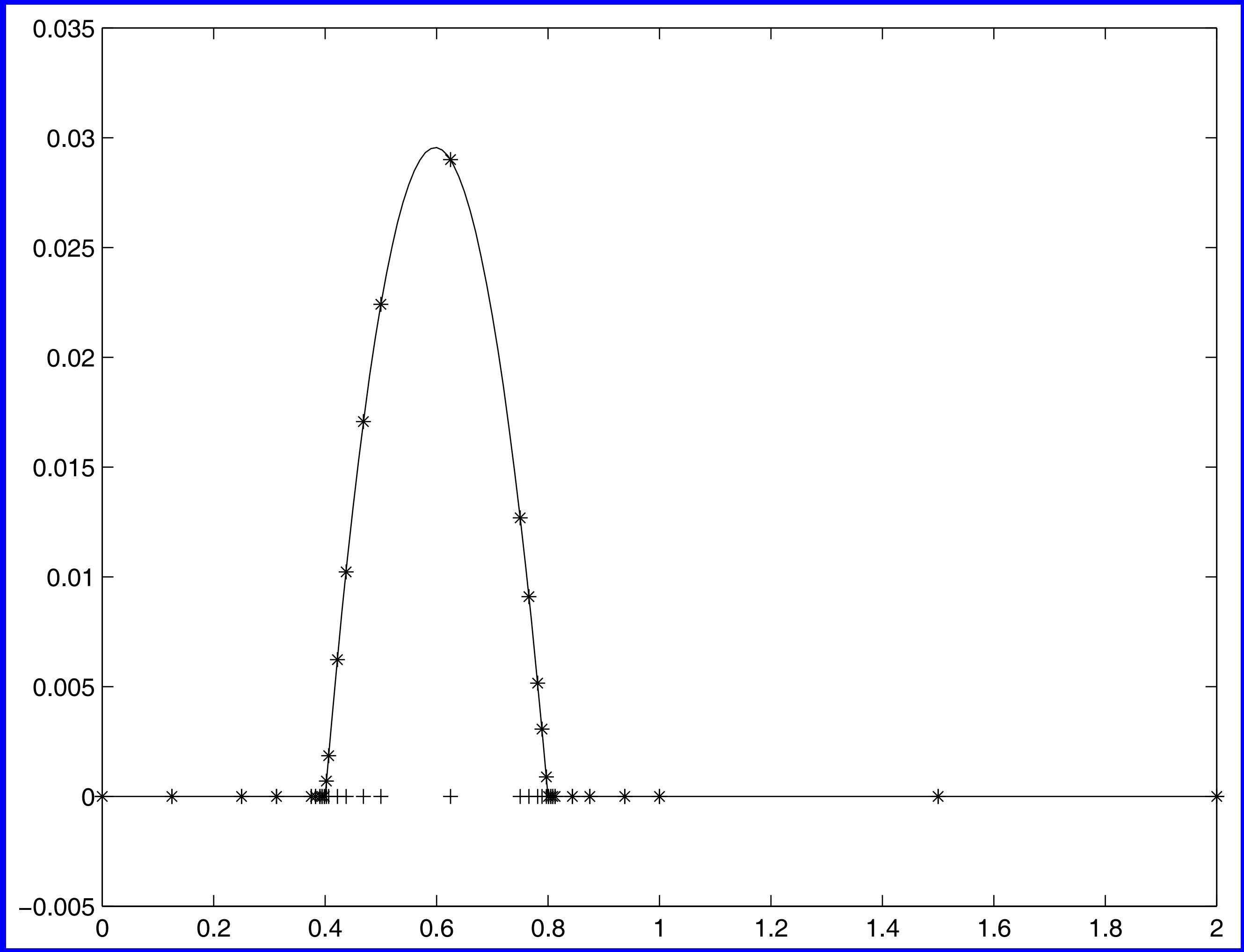
8. Compute β and α_{max}.

9. Compute α_k, using a strategy that significantly reduce the merit function

10. Compute x_{k+1}, s_{k+1} and λ_{k+1}.

11. Go to step 4.

Numeric integration



Numerical results / Conclusions

- *Discretization method*

  ⋆ Solves all problems in the (SIP)AMPL database (over 160 problems) except problems *elke2* and *blankenship2/3*;

  ⋆ Solution found in the finest grid (no KKT point);

  ⋆ Needs NPSOL to solve the finite subproblems.

Numerical results / Conclusions

- *Discretization method*
  - ⋆ Solves all problems in the (SIP)AMPL database (over 160 problems) except problems *elke2* and *blankenship2/3*;
  - ⋆ Solution found in the finest grid (no KKT point);
  - ⋆ Needs NPSOL to solve the finite subproblems.

- *SQP method*
  - ⋆ Solves all problems with only one infinite variable and without finite constraints, except for the robotics problems;
  - ⋆ Needs NPSOL to solve the finite subproblems.

Numerical results (cont.) / Conclusions

- *Penalty method*
  - ⋆ Solves all problems with only one infinite variable and without finite constraints;

Numerical results (cont.) / Conclusions

- *Penalty method*
  - ⋆ Solves all problems with only one infinite variable and without finite constraints;
- *Interior point method*
  - ⋆ Solves 75% of problems with only one infinite variable and without finite constraints.

The End

email: aivaz@dps.uminho.pt
emgpf@dps.uminho.pt
p.gomes@ic.ac.uk
Web http://www.norg.uminho.pt/aivaz/

First Page

Talk partially financed by the Portuguese Calouste Gulbenkian Foundation