Exploiting Structure in Unconstrained Derivative Free Optimization

Philippe Toint
(University of Namur, Belgium)
(with B. Colson)

Amritsar, January 2001

Outline

- The problem and some of its history
- The basis of interpolation methods
  - poisedness
  - geometry improvement
  - a trust-region algorithm
- Geometry and fundamental polynomials
  - Lagrange
  - Newton
  - the interpolation model
  - a glimpse of convergence theory

Outline (2)

- Introducing the sparsity structure
  - choosing the adequate approximation space
  - consequences on storage
  - consequences on iteration costs
- Numerical experience
  - the test functions
  - the results
- Conclusions and perspectives

The problem

Consider the unconstrained problem

$$\min_x f(x)$$

Gradient (and Hessian) of $f(x)$ unavailable

- physical measurement
- object code
- typically small-scale

$\Rightarrow$ "Derivative free optimization" (DFO)

The problem (2)

$f(x)$ typically very costly

Exploit each evaluation of $f(x)$
to the utmost possible

considerable interest of practitioners

Possible approaches

- pattern search methods

  Box ('57), Campey-Nickols ('61),

  Hooke-Jeeves ('61), Spendley-Hext-Himsworth ('62),

  Nelder-Mead ('65), Dixon ('73) [ACSIM],

  Dennis-Torczon ('91) [PDS], Buckley ('94),

  Wright ('96), Torczon ('97), Dennis-Torczon ('97),

  Alexandrov et al. ('98), Audet ('99), . . .

- line search methods

  Powell ('64) [VA04, VA24], Brent ('73) [PRAXIS],

  Toint-Callier ('77), Lucidi-Sciandrone ('96), . . .

Possible approaches (2)

- finite-difference gradients + quasi-Newton

  Powell ('65), Stewart ('67), Powell ('70),

  Gill-Murray-Wright ('81), Dennis-Schnabel ('83),

  . . .

- interpolation methods

Interpolation methods for DFO

The idea:

Until "convergence":

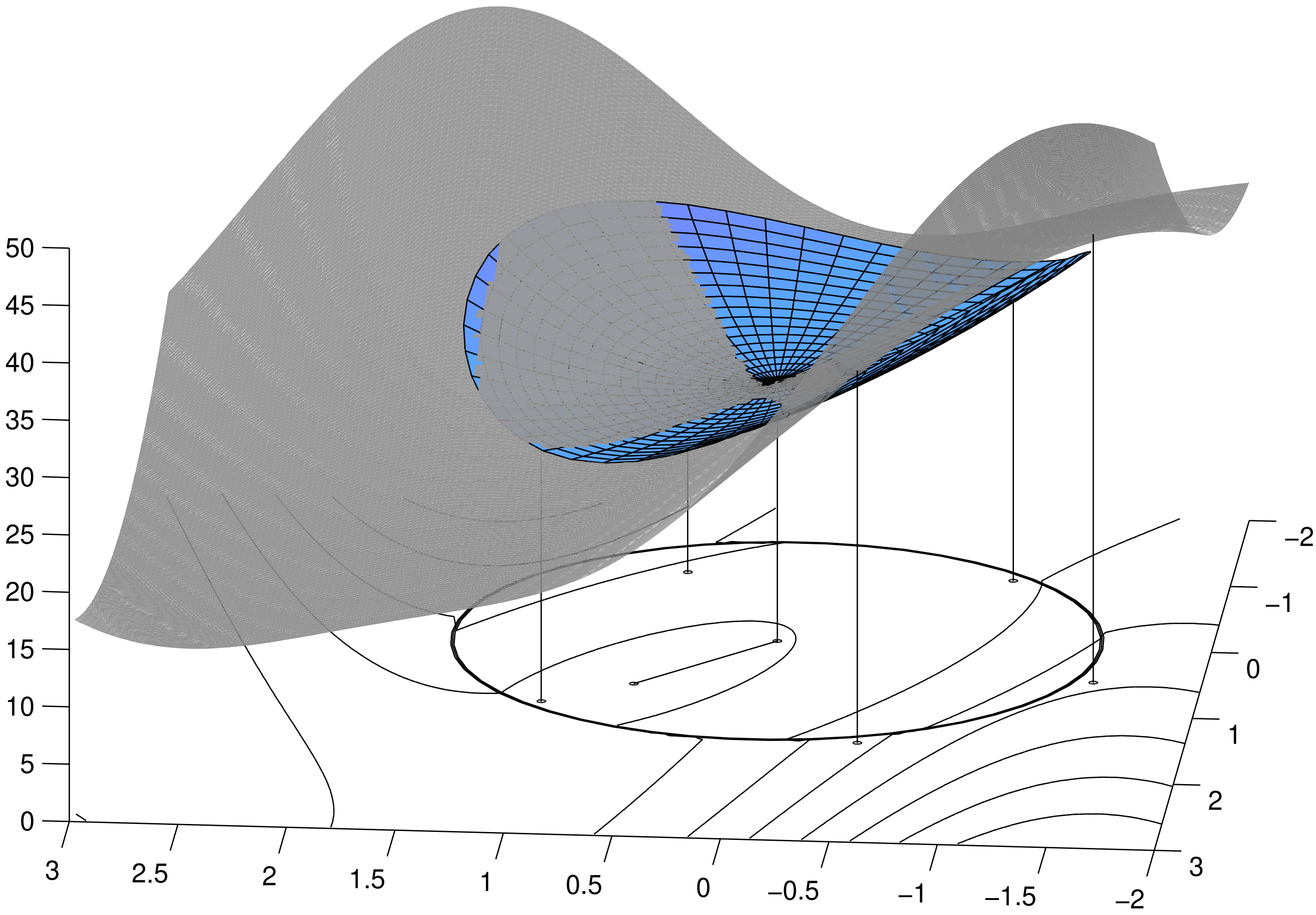
- Use the available function values to build a polynomial interpolation model m_k:

$$m_k(y_i) = f(y_i) \quad y_i \in Y;$$

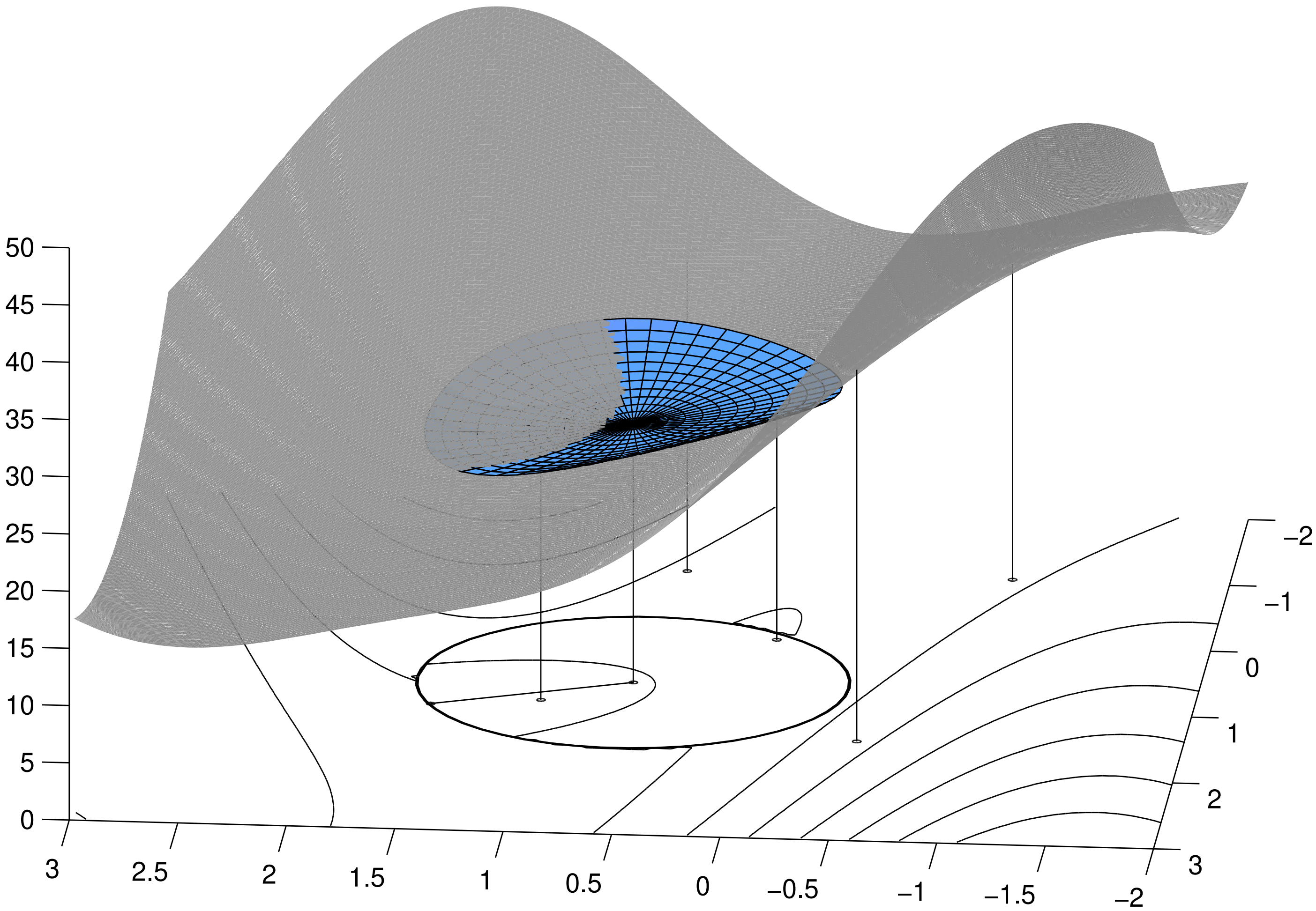
- Minimize the model in a "trust region", yielding a new potentially good point;
- Compute a new function value.

Y = "interpolation set"
$\subseteq$ { points y_i at which $f(y_i)$ is known }

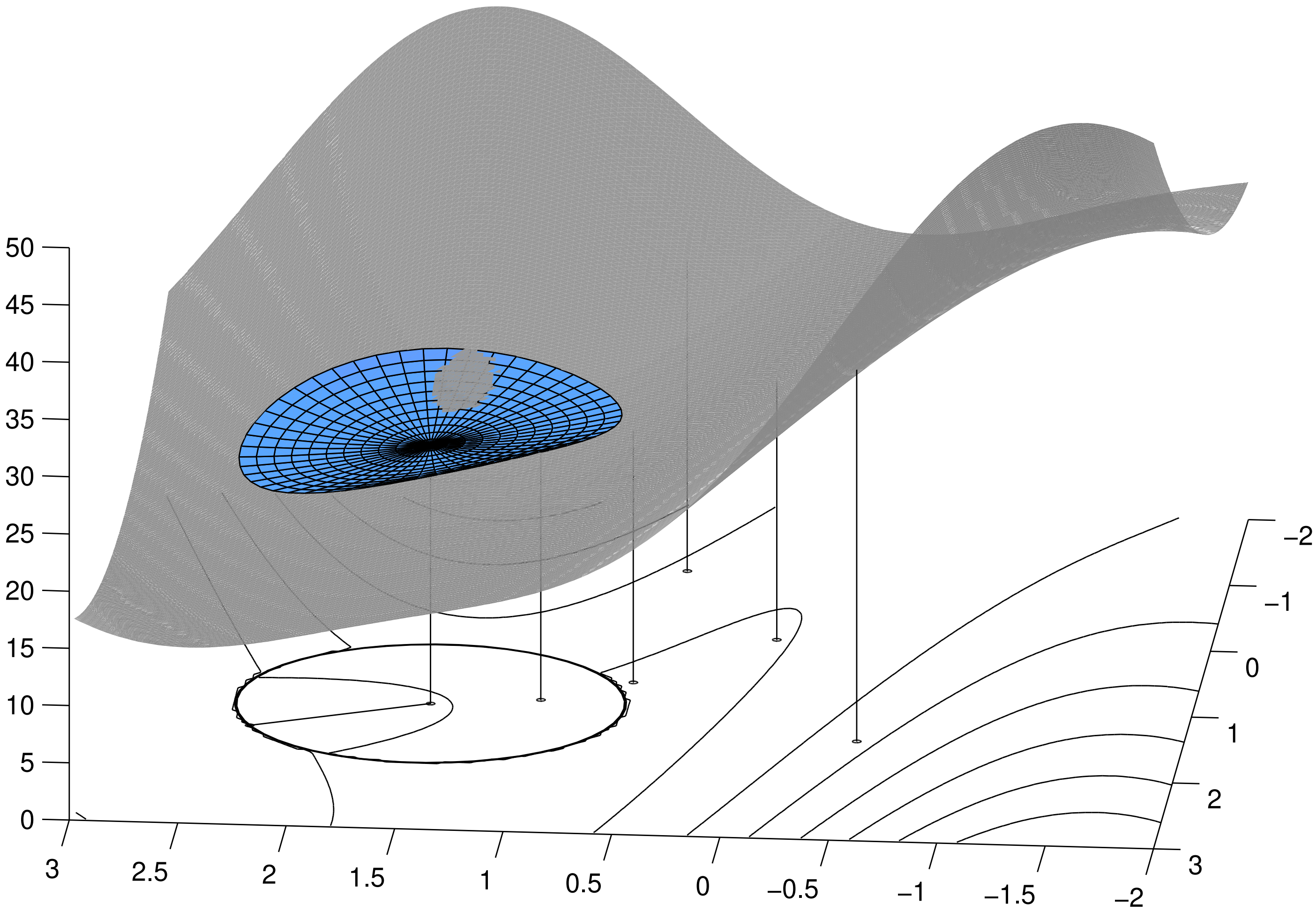
A trust-region method for DFO



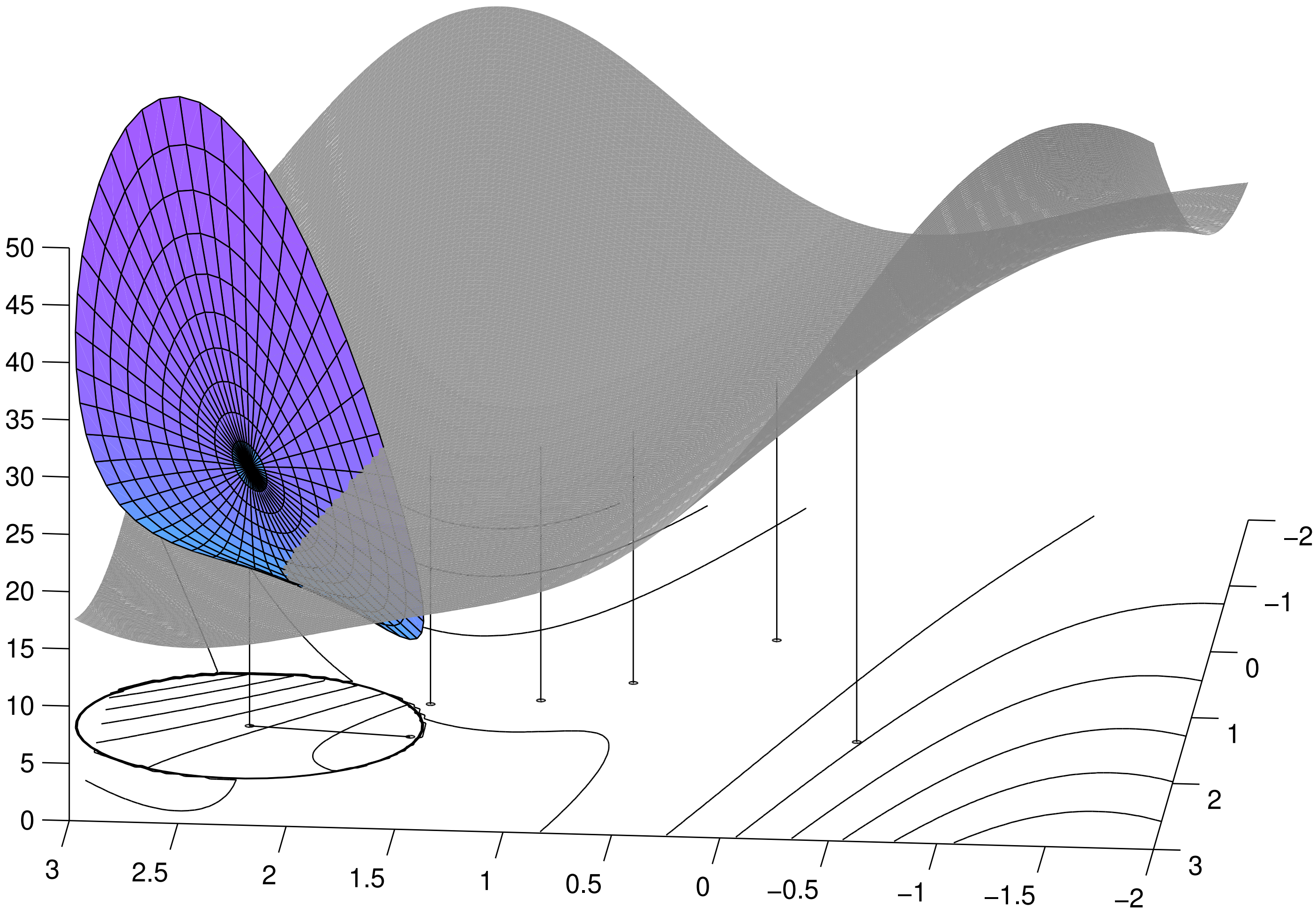
A trust-region method for DFO (2)



A trust-region method for DFO (3)



A trust-region method for DFO (4)



Interpolation methods for DFO (2)

To be considered:

- poisedness of the interpolation set Y
- choice of models
  - approximation subspace
  - choice of basis
- (convergence theory)
- numerical performance

Historical outline for interpolation methods

- early proposal:
  full quadratic models,
  poisedness ???, convergence ???
  Winfield ('69), Winfield ('73)

- revival:
  full or partial quadratic models,
  mechanism to ensure poisedness,
  proven convergence
  Powell ('94), Conn-Toint ('96), Powell ('96),
  Conn-Scheinberg-Toint ('97) ('98),
  Powell ('99), Powell ('00), Marazzi-Nocedal ('00),
  . . .

Poisedness

Assume a quadratic model

$$m_k(x_k + s) = f_k + \langle g_k, s\rangle + \frac{1}{2}\langle s, H_k s\rangle$$

Thus

$$p = 1 + n + \frac{1}{2}n(n+1) = \frac{1}{2}(n+1)(n+2)$$

parameters to determine

$\Rightarrow$ need p function values ($|Y| = p$)

not sufficient!

$\Rightarrow$ need geometric conditions
for the points in Y . . .

Poisedness: geometry

Counter-examples ($n = 2, p = 6$)

- 6 points on a line
- 6 points on a circle
- 6 points on a quadratic curve

$\Rightarrow$ measure of poisedness for Y?

Poisedness: geometry (2)

If $\{\phi_i(\cdot)\}_{i=1}^p$ = basis for approximation space, then

$$\sum_{i=1}^{p} \alpha_i \phi_i(y_j) = f(y_j) \quad j = 1, \ldots, p$$

Possible measure

$$\delta(Y) = \det \begin{pmatrix} \phi_1(y_1) & \cdots & \phi_p(y_1) \\ \vdots & & \vdots \\ \phi_1(y_p) & \cdots & \phi_p(y_p) \end{pmatrix}$$

$$\boxed{Y \text{ (well) poised} \Leftrightarrow |\delta(Y)| \geq \epsilon}$$

- scale for the spread of the y_i's

- notion of geometry improvement

  $\Rightarrow$ more convenient measures

Other algorithmic ingredients

- include a new point in the interpolation set
  - need to drop an existing interpolation point?
  - select which one to drop:
    make Y "as poised as possible"

note:
model/function minimizer may produce bad geometry!!
$\Rightarrow$ geometry improvement procedure . . .

Other algorithmic ingredients (2)

- trust-region radius management
  trust region:

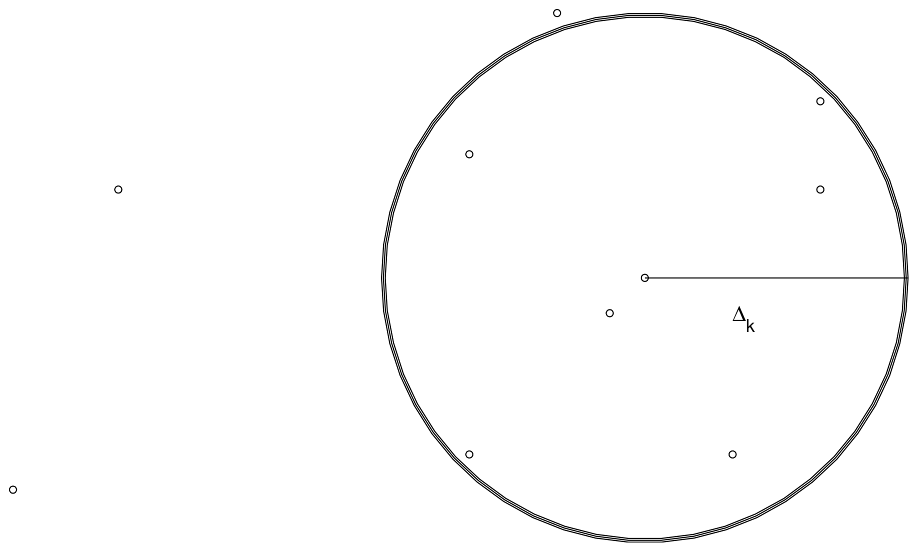
$$\mathcal{B}_k = \{x_k + s \mid \|s\| \leq \Delta_k\}$$

  – classical: reduce Δ_k when "no progress"

  – DFO: more complicated!
    need to verify that Y is poised before reducing Δ_k

  $\Rightarrow$ geometry improvement procedure ...

Improving the geometry in a ball



- attempt to reuse past points
  that are close to x_k

- attempt to replace a distant point of Y

- attempt to replace a close point of Y

if impossible:
good geometry given the current Δ_k

Trust-region algorithm for DFO (1)

criticality test: if g_k small, then

improve the geometry until Y is poised in a ball of radius $\delta \leq \mu \|g_k\|$, centered at x_k

subproblem: compute s_k from

$$\min_{s \in \mathcal{B}_k} m_k(x_k + s)$$

evaluation: compute $f(x_k + s_k)$ and

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}$$

if $\rho_k \geq \eta_1$, define $\mathcal{X}_k = \{x_k + s_k\}$
else $\mathcal{X}_k = \{x_k\}$.

Trust-region algorithm for DFO (2)

model management:

if $\rho_k \geq \eta_1$, include $x_k + s_k$ in Y,
else if Y not poised, improve the geometry
(possibly enlarging $\mathcal{X}_k$)

next iterate:

compute

$$\widehat{x}_k = \arg\min_{x \in \mathcal{X}_k} f(x)$$

$$\widehat{\rho}_k = \frac{f(x_k) - f(\widehat{x}_k)}{m_k(x_k) - m_k(x_k + s_k)}$$

if $\widehat{\rho}_k \geq \eta_0$, accept $\widehat{x}_k$, $(x_{k+1} = \widehat{x}_k)$
else keep x_k $(x_{k+1} = x_k)$.

Trust-region algorithm for DFO (3)

radius update: (remember $\eta_0 < \eta_1$)

if $\widehat{\rho}_k \geq \eta_0$ or Y was poised

$$\hat{\Delta} = \Delta_k$$

if $\rho_k \geq \eta_1$

$$\Delta_{k+1} \geq \hat{\Delta}$$

else

$$\Delta_{k+1} \in [\gamma_1 \hat{\Delta}, \gamma_2 \hat{\Delta}]$$

Using the Lagrange fundamental polynomials

Remarkable: replace y_- by y_+ in Y:

$\dfrac{\delta(Y_+)}{\delta(Y)} = L(y_+, y_-)$ is independent of the base!

where

$$L(y, y_-) = \begin{cases} 1 & \text{if } y = y_- \\ 0 & \text{otherwise} \end{cases}$$

is the Lagrange fundamental polynomial

note: $L(y, y_-)$ is a quadratic polynomial

Using the Newton fundamental polynomials

Choosing a basis for the approximation space: the Newton fundamental polynomials

- points in Y $\longleftrightarrow$ Newton fundamental polynomials

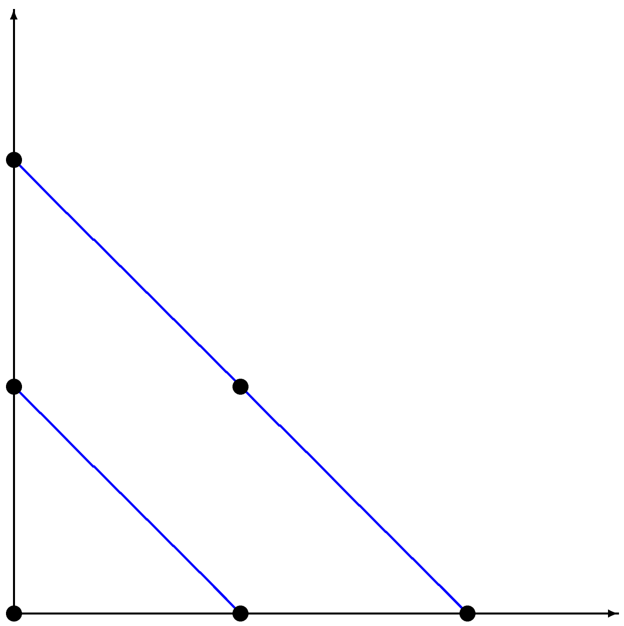
$$\boxed{y_j^{[k]} \in Y \longleftrightarrow N_i^{[\ell]}(x)}$$

- Y structured in $d+1$ blocks $Y^{[\ell]}$: containing polynomials of equal degree

$$\boxed{N_i^{[\ell]}(y_j^{[k]}) = \delta_{ij}\delta_{k\ell} \quad y_j^{[k]}, y_i^{[\ell]} \in Y, \quad k \leq \ell}$$

The Newton fundamental polynomials

Example: 2D interpolation on a regular grid:



$$Y^{[0]} = \{(0,0)\}, \quad Y^{[1]} = \{(1,0),(0,1)\},$$

$$Y^{[2]} = \{(2,0),(1,1),(0,2)\},$$

Initial basis of monomials :

$$\{1, \quad x_1, \quad x_2, \quad x_1^2, \quad x_1x_2, \quad x_2^2\}$$

(for quadratic interpolation)

The Newton fundamental polynomials (2)

$N_j^{[k]}$ obtained by "orthogonalization" of the initial basis

$\Rightarrow$ pivots: $N_i^{[\ell]}$

$\boxed{Y \text{ not poised} \Leftrightarrow \text{zero pivot}}$

Here:

$$N_1^{[0]} = 1, \quad N_1^{[1]} = x_1, \quad N_2^{[1]} = x_2,$$

$$N_1^{[2]} = \frac{1}{2}(x_1^2 - x_1), \quad N_2^{[2]} = x_1 x_2, \quad N_3^{[2]} = \frac{1}{2}(x_2^2 - x_2),$$

The Newton fundamental polynomials (3)

Another example:
interpolate

$$(x_2 - 3\sin x_1)^2 - x_1^2 + \frac{1}{100}(x_1 - 4)^4 + e^{2x_1 - 8}$$

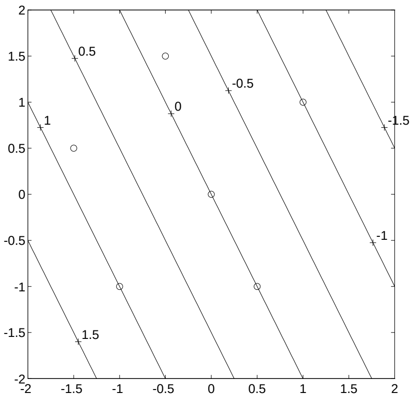
with the monomials basis on

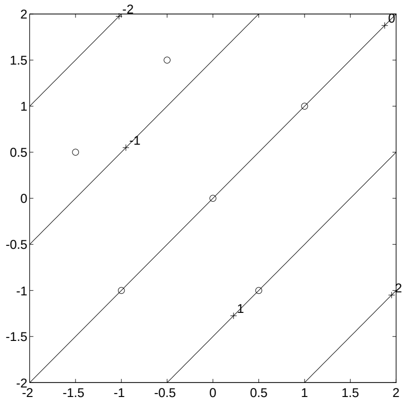
$$Y^{[0]} = \{(0,0)\},$$

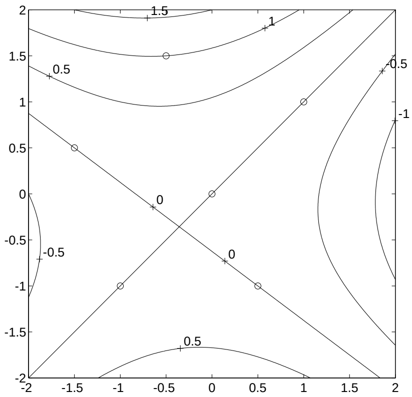
$$Y^{[1]} = \{(-1,-1),\ (\tfrac{1}{2},-1)\},$$

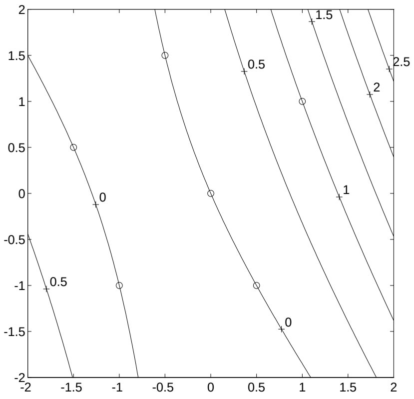
$$Y^{[2]} = \{(-\tfrac{1}{2},\tfrac{3}{2}),\ (1,1),\ (-\tfrac{3}{2},\tfrac{1}{2})\}.$$

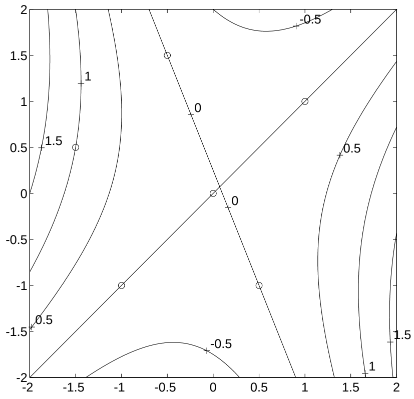
The Newton fundamental polynomials (4)









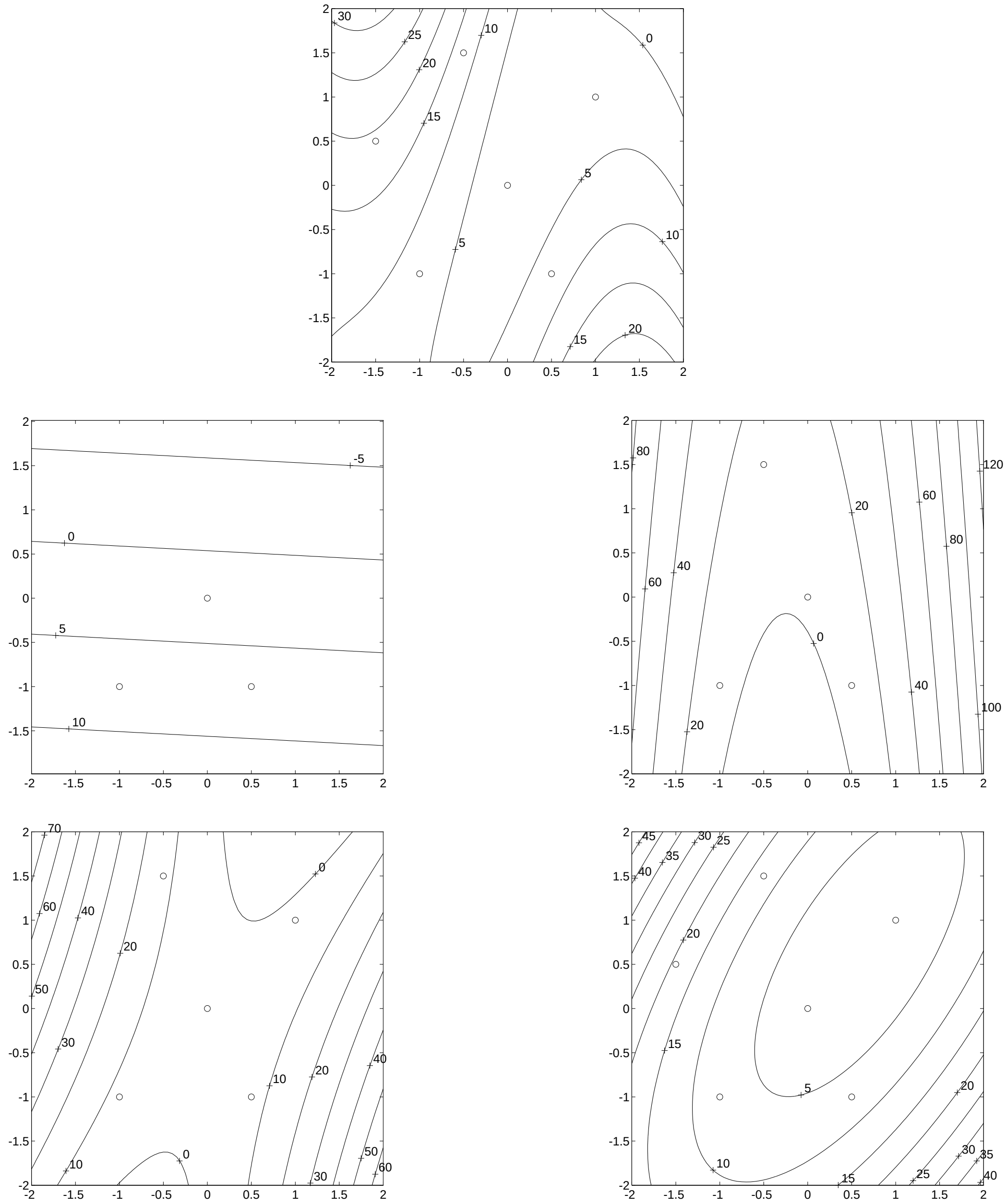


The Newton fundamental polynomials (5)

$$m(x) = \sum_{y_i^{[\ell]} \in Y} \lambda_i^{[\ell]}(Y, f) N_i^{[\ell]}(x)$$

- $\lambda_i^{[\ell]}(Y, f)$ are generalized divided differences
- the sum may be incomplete if Y not poised: consider the subsets
  $\{(0,0),\ (-1,-1),\ (\frac{1}{2},-1)\}$,
  $\{(0,0),\ (-1,-1),\ (\frac{1}{2},-1),\ (-\frac{1}{2},\frac{3}{2})\}$,
  $\{(0,0),\ (-1,-1),\ (\frac{1}{2},-1),\ (-\frac{1}{2},\frac{3}{2}),\ (1,1)\}$
  $\{(0,0),\ (-1,-1),\ (\frac{1}{2},-1),\ (-\frac{1}{2},\frac{3}{2}),(\ 1,1),\ (-\frac{3}{2},\frac{1}{2})\}$.

The Newton fundamental polynomials (6)



A glimpse of the convergence theory

Newton fundamental polynomials
$\Rightarrow$ Bound on the interpolation error
(Sauer-Xu ('95))

$$\boxed{|f(x) - m_k(x)| \le \kappa_2 \Delta^3 \max_{Y,x} |N_i^{[\ell]}(y_j^{[\ell+1]})|}$$

$\Rightarrow$ trust-region convergence theory!

model at least fully linear
interpolation set well poised
$\Rightarrow$ convergence to 1rst-order critical points

model at least fully quadratic
interpolation set well poised
$\Rightarrow$ convergence to 2nd-order critical points

Geometry and Newton polynomials

idea: use orthogonalization pivots

- Strategy for including y_+: replace

  $y_- = y_i^{[\ell]}$ associated with smallest pivot

  ($\Rightarrow$ decrease $|N_i^{[\ell]}|$ and thus $|N_i^{[\ell]}(y_j^{[\ell+1]})|$)

- strategy for improving the geometry:
  replace y_- by

  $$y_+ = \arg\max_{y \in \mathcal{B}} |N_i^{[\ell]}(y)|$$

  if $|N_i^{[\ell]}(y_+)| \geq 1 + \epsilon$
  (otherwise the geometry is fine)

Introducing sparsity

Assume $\nabla_{xx} f(x)$ is known to be $\boxed{\text{sparse}}$ (independently of x)

$\Rightarrow$ exists $\mathcal{S} = \{(i,j) | 1 \leq i,j \leq n\}$ (symmetric)

with coefficient of $x_i x_j$ identically zero in Taylor's expansion of f, i.e.

$$\forall x \quad \forall (i,j) \in \mathcal{S} \quad \langle e_i, \nabla_{xx} f(x) e_j \rangle = 0 \qquad (*)$$

Simple idea:

reduce the approximation space to the set of quadratic polynomials satisfying (*)

Introducing sparsity (2)

$\Rightarrow N_j^{[k]}$ obtained by orthogonalization of a partial basis (excluding $x_i x_j$ for $(i,j) \in \mathcal{S}$)

$\Rightarrow$ fewer interpolation points needed (Y smaller)

$$m_k(x_k + s) = \sum_{y_i^{[\ell]} \in Y} \lambda_i^{[\ell]}(Y, f) N_i^{[\ell]}(s)$$

$\boxed{\text{Triple impact:}}$

- less data points needed for complete interpolation
- less polynomials need being built and stored
- each polynomial smaller

Numerical experience

More general linear constraints

Conclusions

- very useful technique when sparsity is known
  - storage
  - speed
  - dominant parts of the Hessian can be used
- introduce "artificial sparsity" ?
- partially separable structure more powerful when known (one model per element)
- applicable to more general linear constraints
- code (UDFO) soon available