University of Ljubljana, Dept. of Physics

Extraction and analysis algorithms

Andrej Prša

8th RVS workshop

June 3-4, 2004 - Padova, Italy

CONTENTS

INTRODUCTION: THE PROBLEM

This study discusses the methods and algorithms involved in the computational solution of:

Unconstrained minimization of multidimensional functions without derivatives

The function (usually referred to as object function): $f : \mathbb{R}^n \longrightarrow \mathbb{R}$

Examples of the object function:

- light curve flux at the given phase from *n* physical parameters
- χ^2 value of the synthetic– observed stellar spectrum (T, log g, [M/H], v)

Minimization:

$$\text{For a given } f : \mathbb{R}^n \longrightarrow \mathbb{R} \text{ we seek } \mathbf{x_0} \in \mathbb{R}^n : f(\mathbf{x_0}) \leq f(\mathbf{x}) \; \forall\, \mathbf{x} \in \mathbb{R}^n$$

Or, in short:

$$\mathbf{x_0} = \min_{\mathbf{x}\in\mathbb{R}^n} f : \mathbb{R}^n \longrightarrow \mathbb{R}$$

INTRODUCTION: THE PROBLEM

This study discusses the methods and algorithms involved in the computational solution of:

Unconstrained minimization of multidimensional functions without derivatives

Unconstrained:

$$\mathbf{x}_0 = \min_{\mathbf{x}\in\mathbb{R}^n} f : \mathbb{R}^n \longrightarrow \mathbb{R}$$

Constrained:

$$\mathbf{x}_0 = \min_{\mathbf{x}\in\mathbf{\Omega}\subseteq\mathbb{R}^n} f : \mathbb{R}^n \longrightarrow \mathbb{R}$$

"Without derivatives" doesn't mean that the derivatives don't exist or that they are not piecewise connected, it only means that we cannot (or would not) use them explicitly.

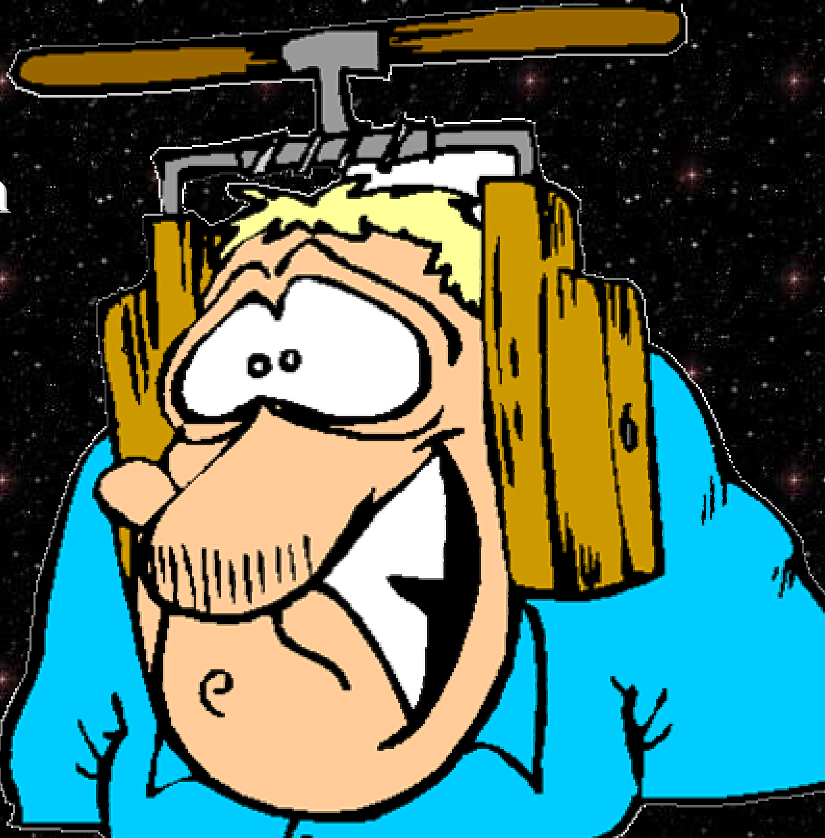
All such methods are globally convergent if any minimum (local or global) is contained within the observed hyperspace.

INTRODUCTION: DEGENERACY, FINDING GLOBAL MINIMUM

Although convergent, these methods never offer any notion of which minimum they converge to. Thus we need heuristic approaches to resolve this problem:

a) use the method on many starting points in hyperspace and compare the different values of convergence. The lowermost will likely be the global minimum.

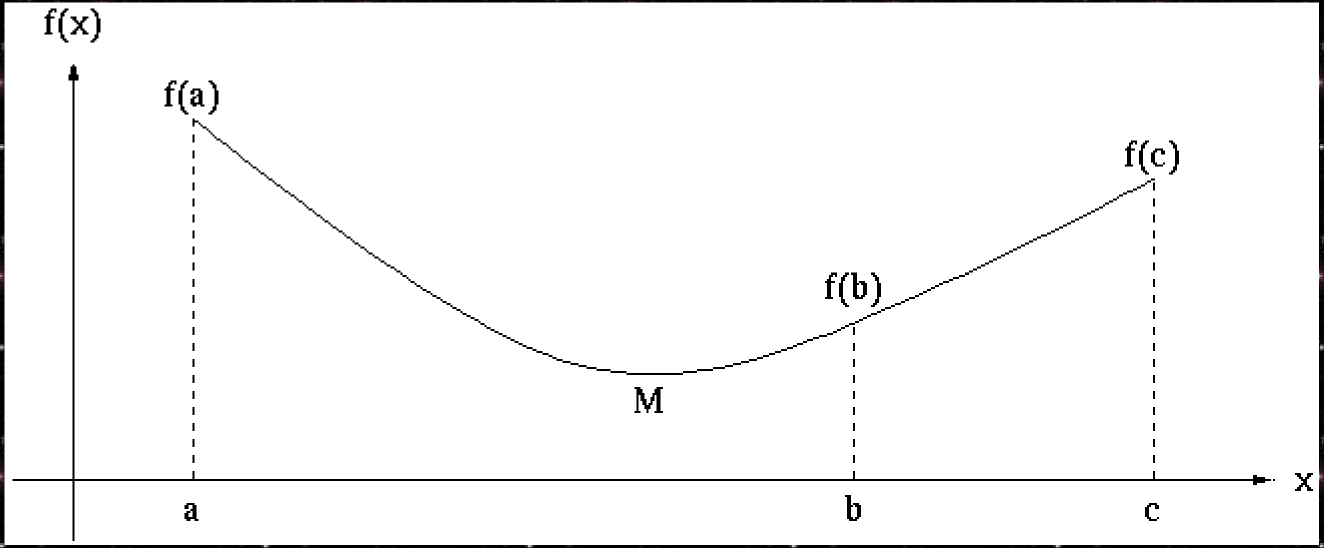
b) once in a given minimum, perturb it by taking a finite step away in some random direction. If for all perturbations the minimum is the same or its value is increased, it is again likely to be the global minimum.



Obviously, right combination of the wrong parameters can give us severe headache!

1st algorithm: BRACKETING OF 1D FUNCTION (BRENT)

For bracketing a minimum in 1D, we need a triplet of points: (a, b, c).



If $a < b < c$ and the following relation between their functional values holds:

$$f(b) < f(a) \quad \wedge \quad f(b) < f(c)$$

then there is a minimum which we may find e.g. by bisection.

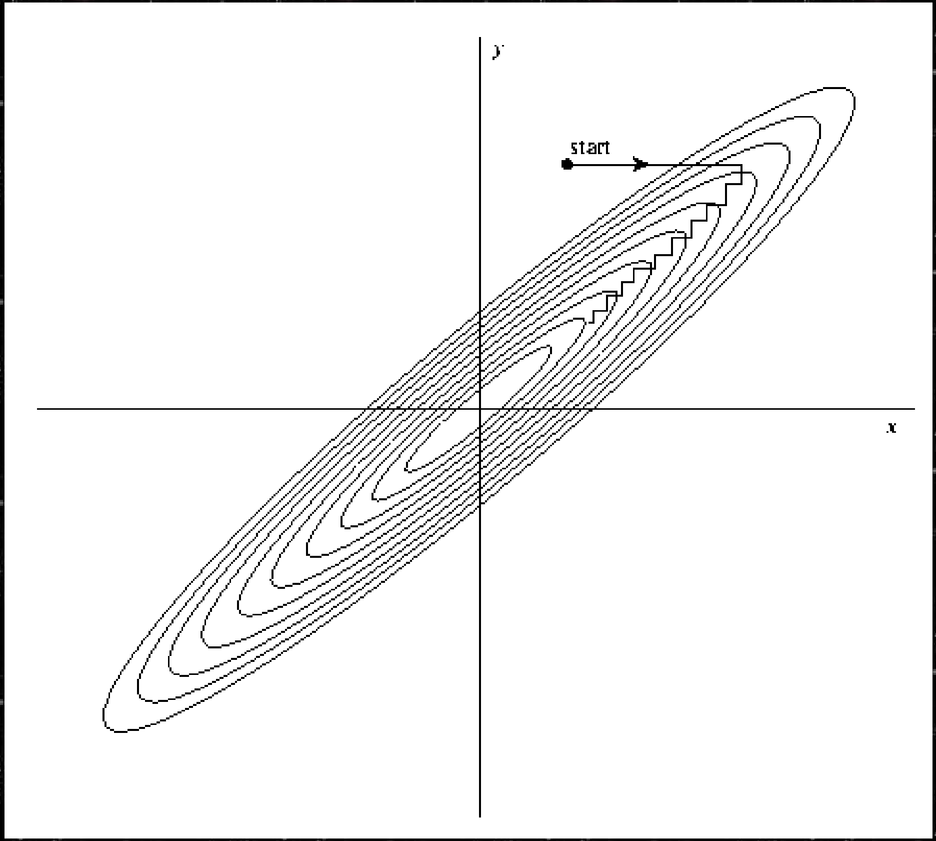
Bracketing a minimum of a given multidimensional function is not possible!

2nd algorithm: POWELL'S DIRECTION SET METHOD

1D bracketing (A.K.A. line bracketing) may still be used for multidimensional functions.

If we start from point ***p*** in our nD hyperspace and proceed from there in some vector direction ***u***, we may minimize our function *f* along the line determined by ***u*** with simple line bracketing.

Once the 1D minimization along ***u*** is complete, the next direction is chosen and the process is repeated.



There are many versions of the Powell's method. They all share the same base, the only difference is in the way they calculate the consecutive set of directions the minimizer should take.

2nd algorithm: POWELL'S DIRECTION SET METHOD

If the function gradient is unattainable, the simplest approach is to follow the unit vectors $\boldsymbol{e}_1$, $\boldsymbol{e}_2$, ..., $\boldsymbol{e}_n$ in the direction set. The algorithm then cycles through all these directions until the function stops decreasing. However, this method may be very inefficient (e.g. for long narrow valleys).

a) come up with a set which includes some very good directions,

b) establish a subset of "non-interfering" directions, that are independent among themselves, i.e. the line minimization along one direction isn't "spoiled" by the subsequent minimization along the other.

- If f is minimized along $\boldsymbol{u}$, then ∇f must be perpendicular to $\boldsymbol{u}$ at minimum.
- The function may be expanded in Taylor series around the origin $\boldsymbol{p}$:

$$f(\mathbf{x}) = f(\mathbf{p}) + \sum_i \frac{\partial f}{\partial x_i} x_i + \frac{1}{2} \sum_{i,j} \frac{\partial^2 f}{\partial x_i \partial x_j} x_i x_j + \ldots = c - \mathbf{b} \cdot \mathbf{x} + \frac{1}{2} \mathbf{x} \cdot \mathbf{H} \cdot \mathbf{x}$$

2nd algorithm: POWELL'S DIRECTION SET METHOD

By taking the gradient of the Taylor expansion:

$$\nabla f = -\mathbf{b} + \mathbf{H} \cdot \mathbf{x}$$

we may calculate the change in gradient when moving along one direction:

$$\delta\left(\nabla f\right) = \mathbf{H} \cdot (\delta \mathbf{x})$$

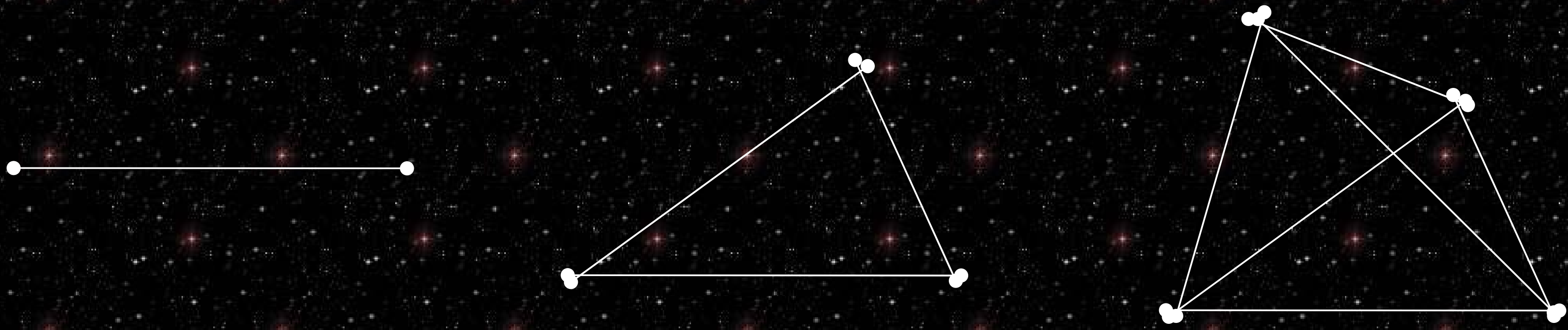
After *f* is minimized along ***u***, the algorithm proposes a new direction ***v***, so that minimization along ***v*** doesn't spoil the minimum along ***u***. For this to be true, the function gradient must stay perpendicular to ***u***:

$$\mathbf{u} \cdot \delta\left(\nabla f\right) = 0 = \mathbf{u} \cdot \mathbf{H} \cdot \mathbf{v}$$

When this is true, u and v are said to be conjugate. For conjugate directions the cycling doesn't make sense and we get quadratic convergence to the minimum!

3rd algorithm: DOWNHILL SIMPLEX METHOD (NELDER & MEAD)

A simplex is an nD geometrical body that is determined by (n+1) vertices along with their interconnecting lines and polygonal faces.



If any of (n+1) vertices are taken as origin, then the rest n vertices span an nD vector (parameter) space.

The essence:

1) The function f is evaluated in each vertex of the simplex.
2) The vertex with the largest value is replaced by the new vertex that i
   obtained by reflection, expansion or contraction (amoebic adaptation
3) If this lowers the function value, the new vertex is adopted.

4th algorithm: SIMULATED ANNEALING

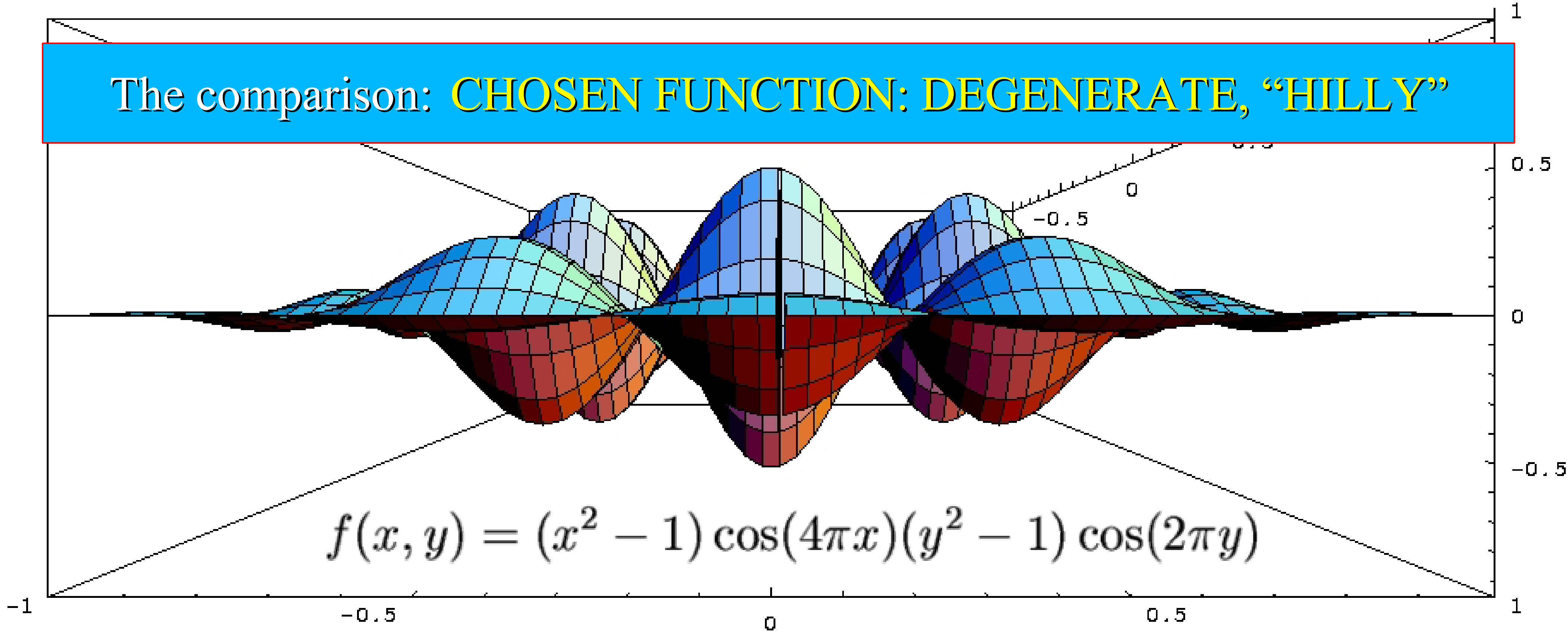
Based on thermodynamical relaxation, suitable for converging to a global mi-nimum among many local minima.

$$p = e^{-(E_i)/(kT)}$$

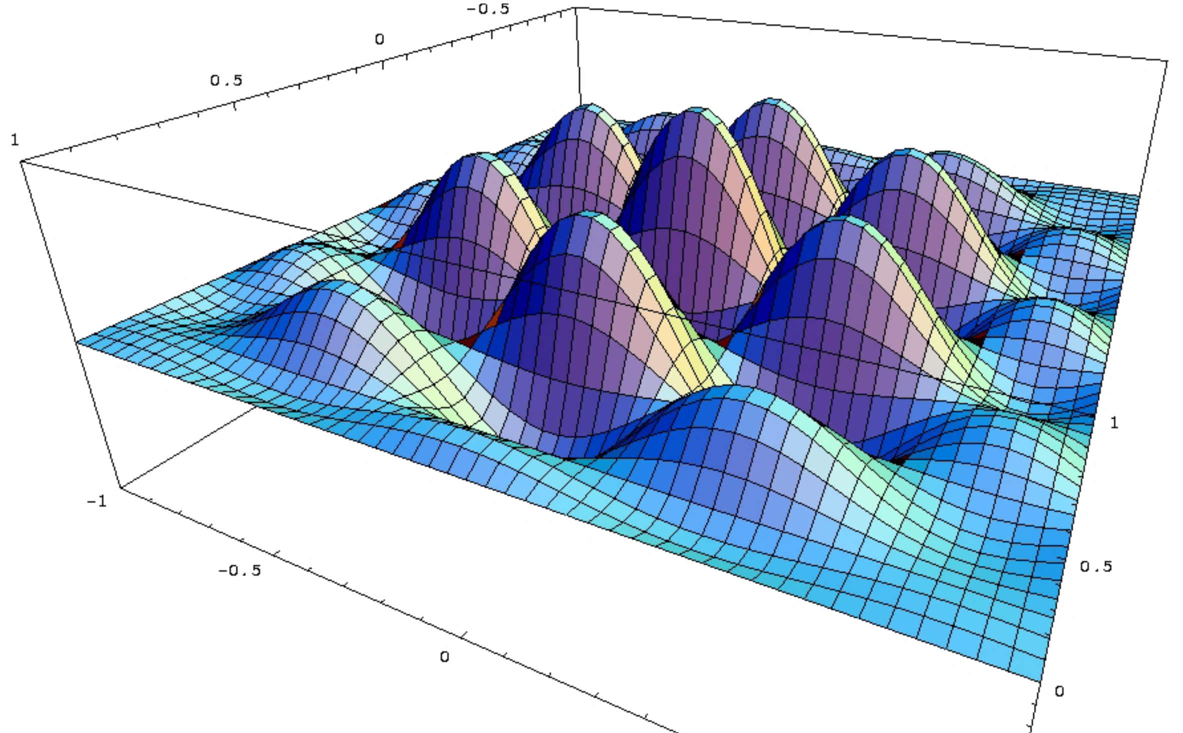
The essence:

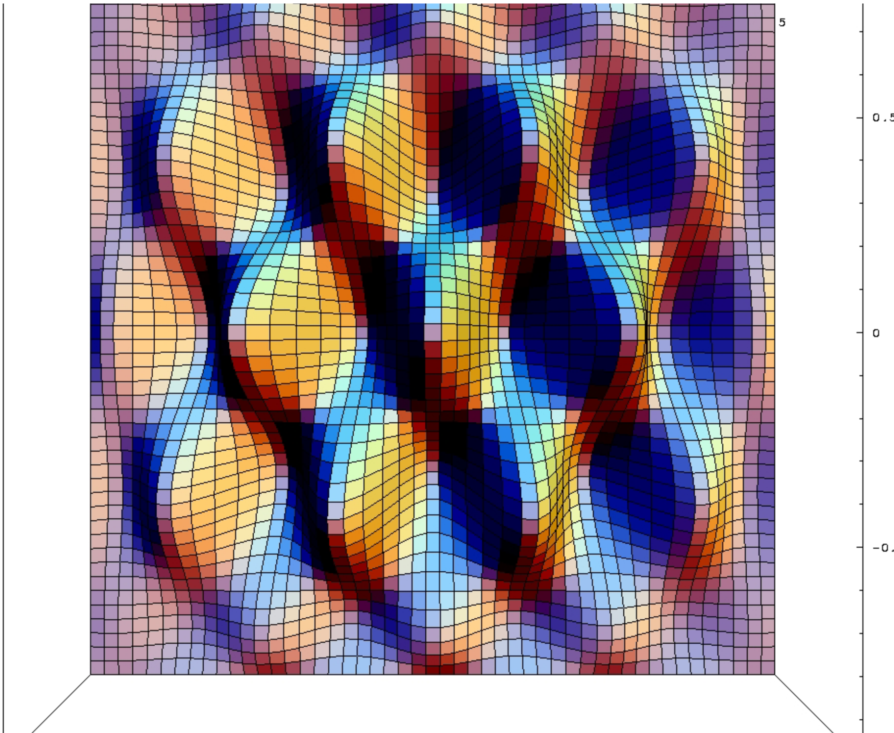
1) The system selects a given (usually arbitrary) initial state.
2) The cost function is calculated (e.g. the function value, χ^2, ...) for that state.
3) A random-walk modification is introduced to the system and the cost function is re-evaluated.
4a) If the new state improves the initial one, it is unconditionally accepted.
4b) If the new state doesn't imrove the initial one, it is discarded with a certain (exponential) probability.

The comparison: CHOSEN FUNCTION: DEGENERATE, “HILLY”



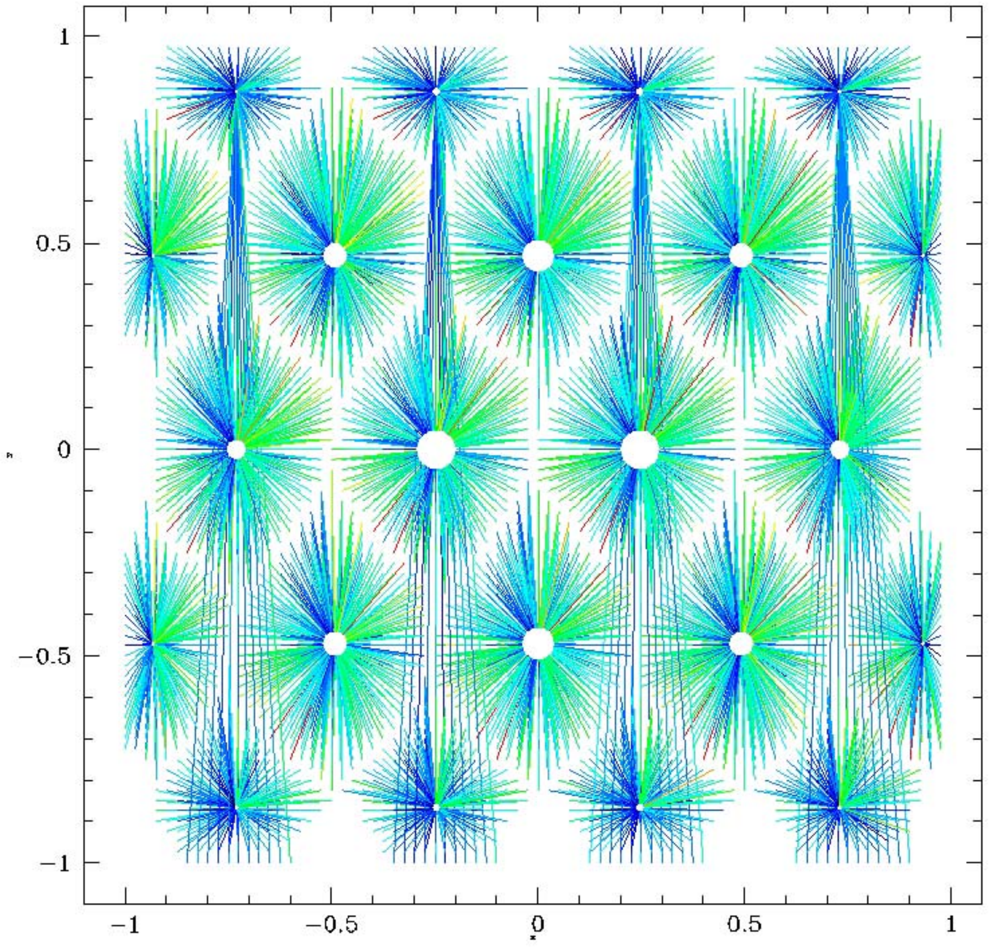
$$f(x,y) = (x^2-1)\cos(4\pi x)(y^2-1)\cos(2\pi y)$$

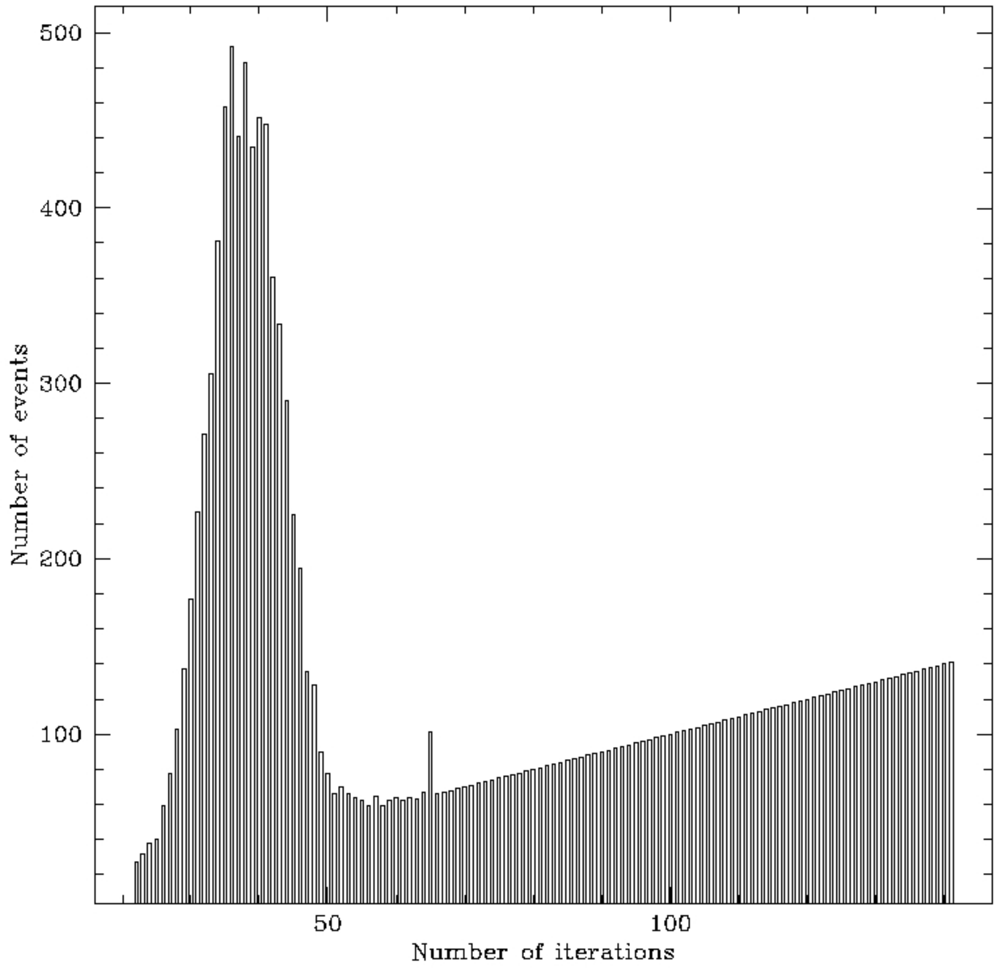




The comparison: DIRECTION SET vs. DOWNHILL SIMPLEX

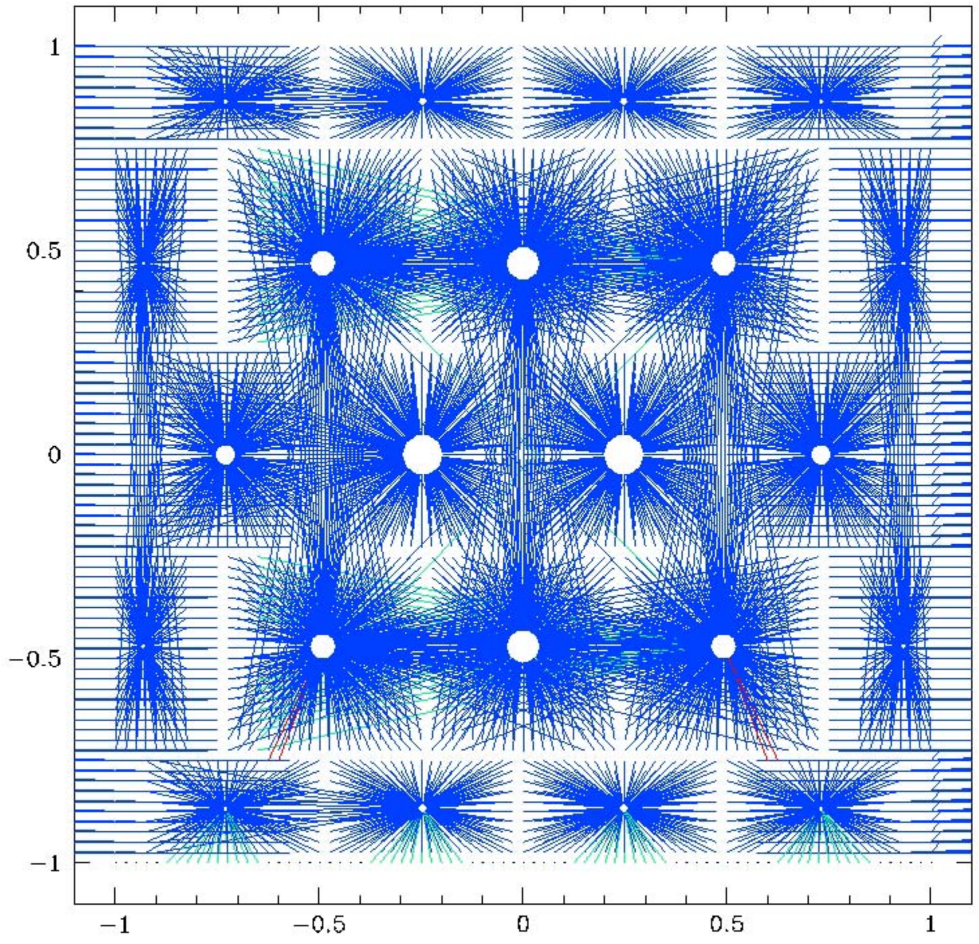
Downhill simplex requires ~35 iterations to reach a (local or global) minimum.

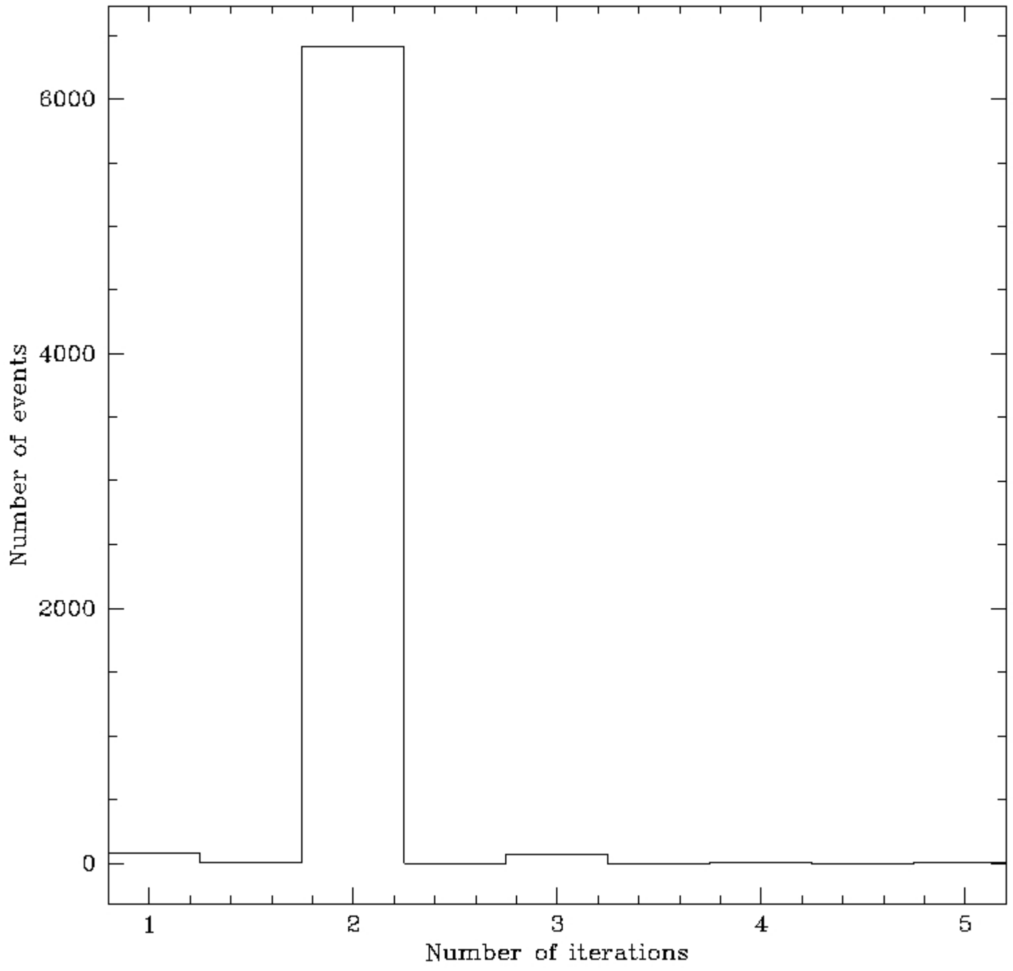




The comparison: DIRECTION SET vs. DOWNHILL SIMPLEX

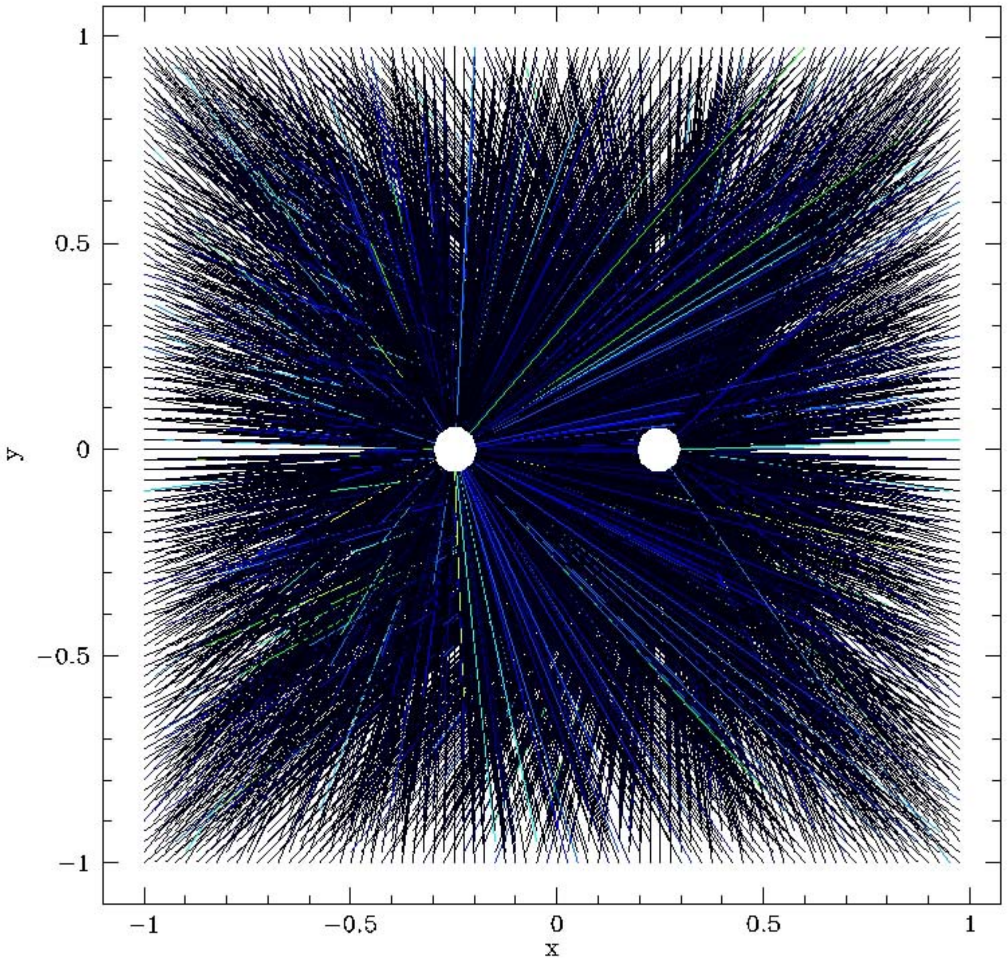
Direction set requires ~2 iterations to reach a (local or global) minimum!

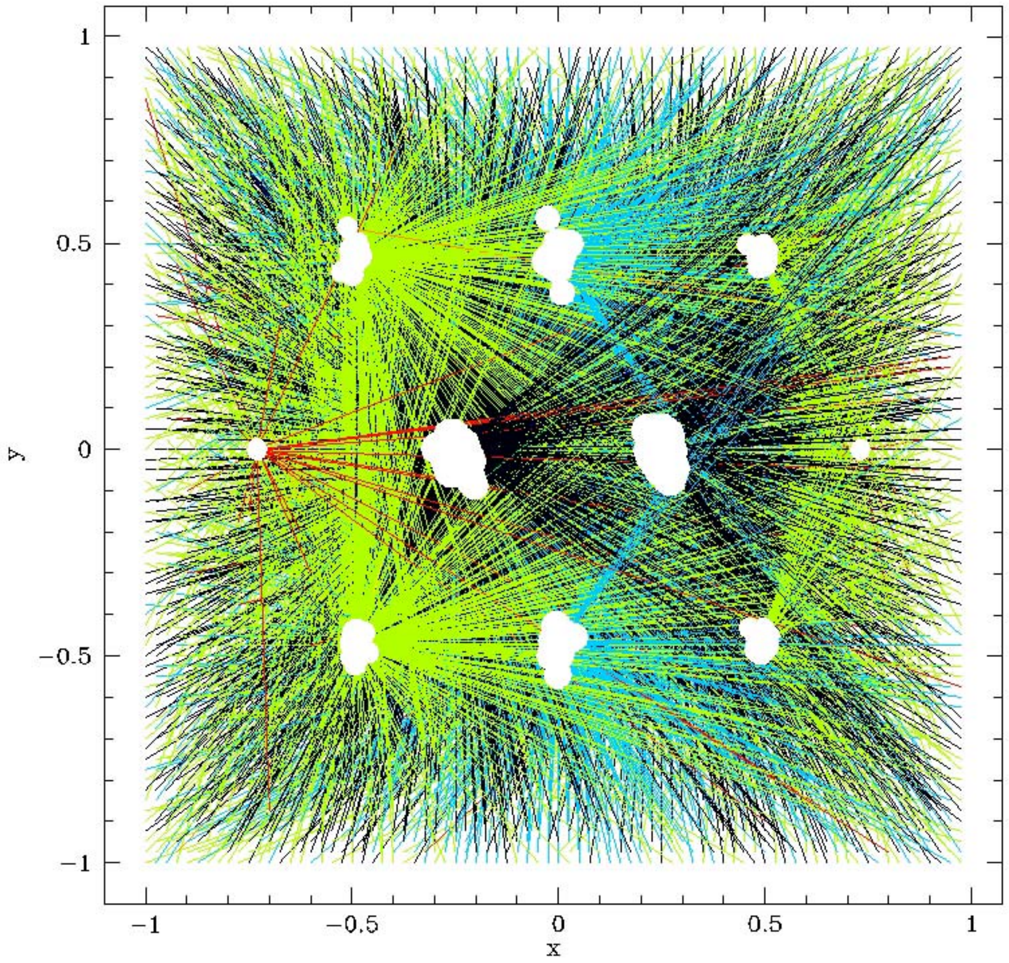




The comparison: DIRECTION SET vs. DOWNHILL SIMPLEX

Simulated annealing after 100 000 (left) and 1 000 (right) cooling steps.





Other ways: GRADIENTS, DC, LEVENBERG-MARQUARDT, ...

1) Numerical methods using gradients
   - 1a) The method of steepest descent (direction set modificatio
   - 1b) Variable metric method (most effective to date)

2) Differential corrections
   - 2a) Finite differences (already implemented, often divergent)
   - 2b) Singular Value Decomposition (SVD)

3) The Levenberg-Marquardt algorithm:
   combines the method of steepest descent with inverse Hessian metho

4) Constrained multidimensional solvers
   linear and non-linear programming (simplex methods)

Classification algorithms: SERIES COEFFICIENTS CORRELATION

Used in most scanning missions of today (ASAS, TASS, OGLE, EROS, ...)

$$f(x) = \sum_{k=1}^{4} a_k \sin(\omega x) + b_k \cos(\omega x)$$

We are observing the correlations between different linear combinations of obtained coefficients.

- Every type is represented by a particular subspace of the whole hyperspace.
- For each cross-plane we calculate the score: 1 within the subspace and exponentially falling out of bounds.
- Multiply scores for all different cross-planes. The highest value wins!
- If there are more winners, double classification is possible.
- If no score is high enough, it is left unclassified.

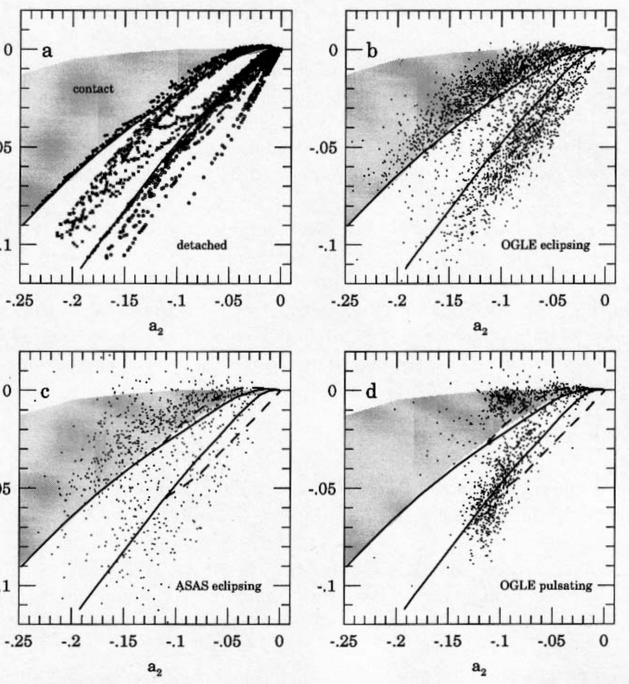


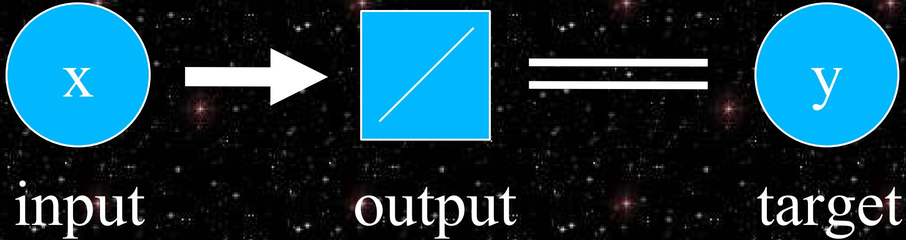
Fig. 4. Distribution of light curves in the a_2 - a_4 plane. a) Data obtained from the Wilson-Devinney simulation are plotted. Grey area encloses contact models calculated by Ruciński (1993). Two continuous lines delineate contact and detached configurations, while semi-detached systems are located between dashed lines. b) Eclipsing variables from the OGLE Bulge database (Wozniak 2002a). c) ASAS eclipsing binaries (this paper). d) OGLE Bulge pulsating variables.

Classification algorithms: NEURAL NETWORKS, BAYES

Before using neural networks, it is crucial to assess whether non-linear regression can solve our problem, since it converges ~1 order faster than NN!

The basic idea of neural networks is taken from biology – there is (a weak) analogy with neurons and synapses.

Symbols and jargon:

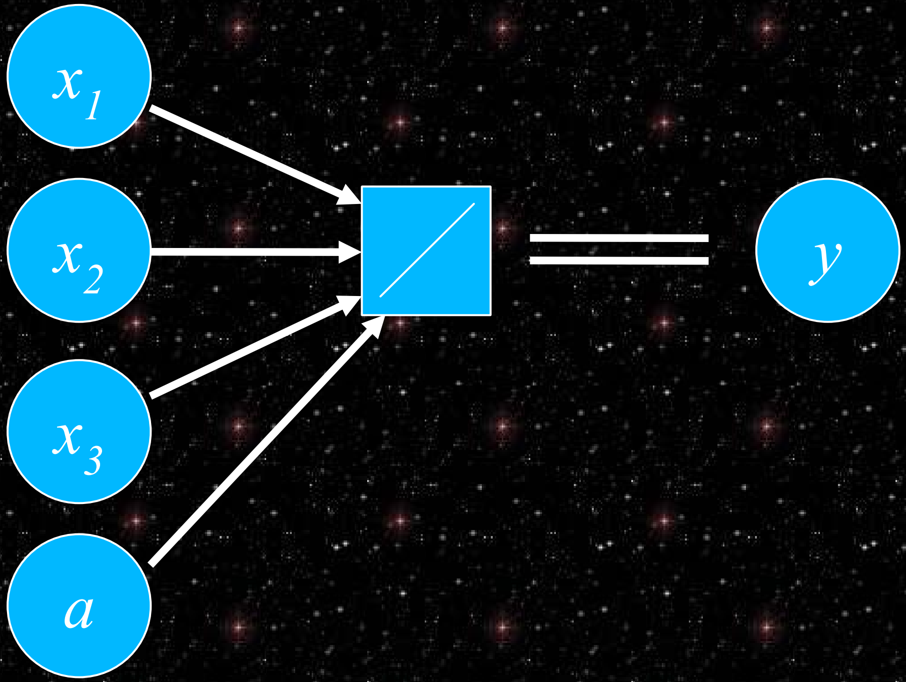


●	observed variables		
■	computed values as a function of one or more variables		linear combination
➡	the source of the arrow is an argument of its destination		logistic function
═	fit both sides by least squares		binary function

Classification algorithms: NEURAL NETWORKS, BAYES

n_x	...	number of inputs
x_i	...	inputs
a_j	...	bias for output layer
b_{ij}	...	weight from input
q_j	...	linear combination
p_j	...	activation function
r_j	...	the residuals

A role of the **perceptron** is to compute q_j (the **net input**). To net input an **activation function** is applied to obtain the output.



e.g. $\mathrm{act}(x) = x$

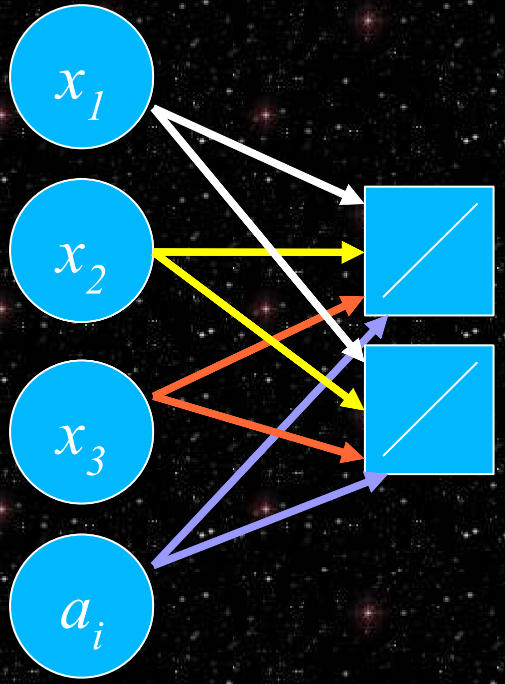
$\mathrm{act}(x) = \left(1 - e^{-x}\right)^{-1}$

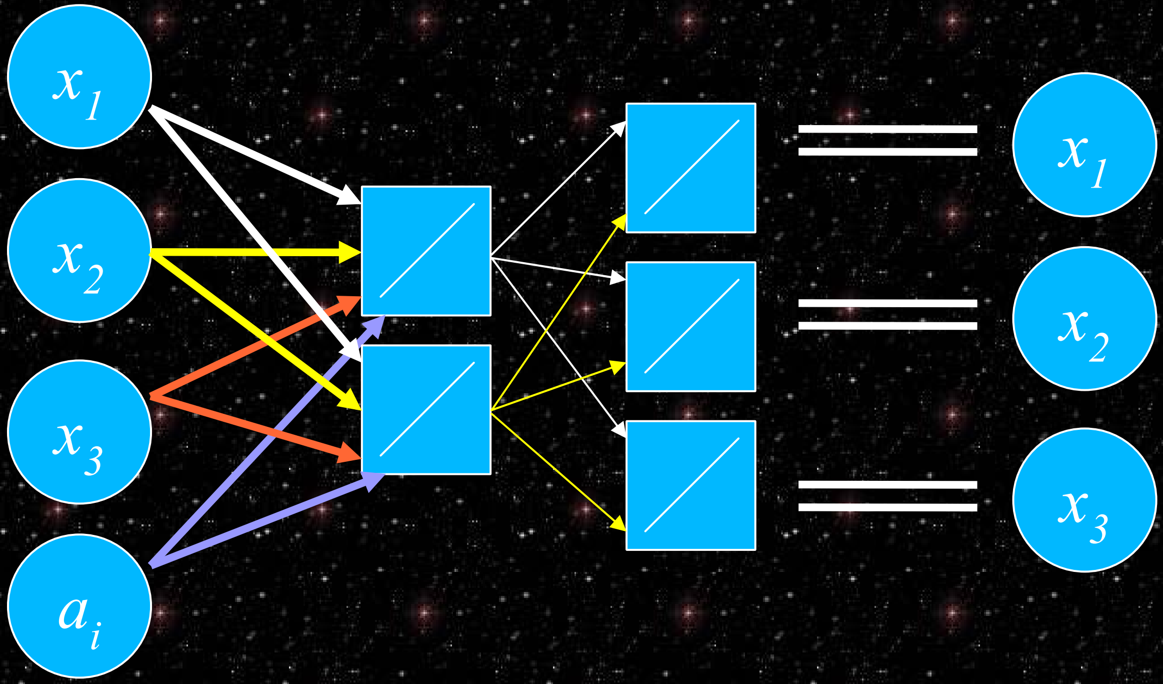
For this particular layer:

$$n_x = 3, \quad i = 1 \ldots 3, \quad a_j = a, \quad q_j = a_j + \sum_{i=1}^{n_x=3} b_{ij} x_j, \quad p_j = \mathrm{act}(x), \quad r_j = p_j - y_j.$$

Classification algorithms: NEURAL NETWORKS, BAYES

For large datasets (such as the ones of GAIA) we need unsupervised learning:





This time, the second layer (the visible one) is a binary layer: only one is activated and all others are suppressed to 0 (winner-take-all model). The winner is the one neuron with the largest net input: the one whose weights are most similar to inputs.

CONCLUSION

There are still other algorithms yet to be investigated and benchmarked for usage plausibility of GAIA data.

Based on our current tests:

- For modeling, Powell's conjugate direction set method with heuristic search seems most attractive, but yet to be proven on real data. [*A working implementation is already used for EBs (PHOEBE)*]

- For classification, Artificial Neural Network approach seems promising, but due to its unphysical approach further testing and benchmarking has to be performed.

Further assessment is needed, we are open for suggestions!

University of Ljubljana, Dept. of Physics

Thank you for your attention!

8th RVS workshop

June 3-4, 2004 - Padova, Italy