Optimization, Engineering & Cluster Computing

Tamara G. Kolda
Sandia National Labs
Livermore, CA

This work was supported by the U.S. Department of Energy.

"It has often been said that a person does not really understand something until he teaches it to someone else. Actually a person does not really understand something until he can teach it to a *computer*, i.e., expresses it as an algorithm."

— Donald E. Knuth, 1974,
Computer Science and Its Relation to Mathematics,
American Mathematical Monthly.

Optimization: Newton's Method

1. Construct a *quadratic model* about iterate x_k:

$$f(x_k + s) \approx f(x_k) + g_k^T s + \frac{1}{2}\, s^T H_k s$$

   where $g_k = \nabla f(x_k)$ and $H_k = \nabla^2 f(x_k)$.

2. Use the model to predict a good *descent direction*:

$$s_k = -H_k^{-1} g_k$$

3. Perform a *line search* to determine the next iterate:

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

There's 30+ years of theory behind these methods...

...but Newton's method doesn't always work

Newton and quasi-Newton methods depend on access to gradient information, but for many engineering problems, we do not have access to such information! The types of problems we are interested in are characterized as follows:

- The evaluation of f is time-consuming,
- the gradient cannot be calculated, and
- the function values are not accurate enough for finite-difference gradient approximations.

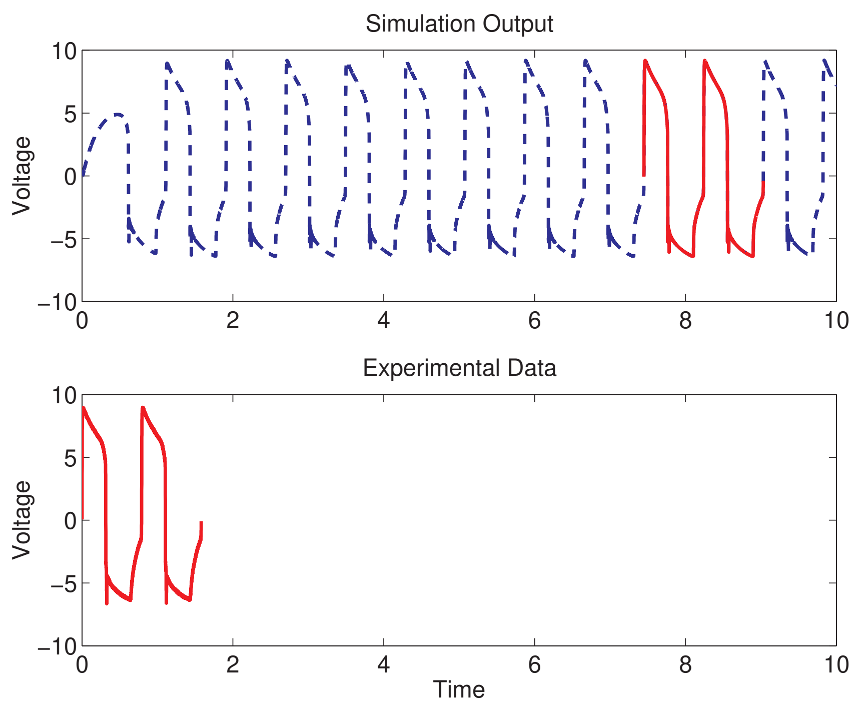
Engineering Example: Determining the Characteristics of a Circuit

- **Variables:** inductances, capacitances, diode saturation currents, transistor gains, leakage inductances, and transformer core parameters
- **Simulation Code:** SPICE3

$$f(x) = \sum_{t=1}^{N} \left(V_t^{\mathrm{SIM}}(x) - V_t^{\mathrm{EXP}}\right)^2,$$

$$\begin{aligned} x &= \text{17 unknown characteristics} \\ V_t^{\mathrm{SIM}}(x) &= \text{Simulation voltage at time } t \\ V_t^{\mathrm{EXP}} &= \text{Experimental voltage at time } t \\ N &= \text{Number of timesteps} \end{aligned}$$

Circuit Simulation



Direct Search
What's Old is New Again

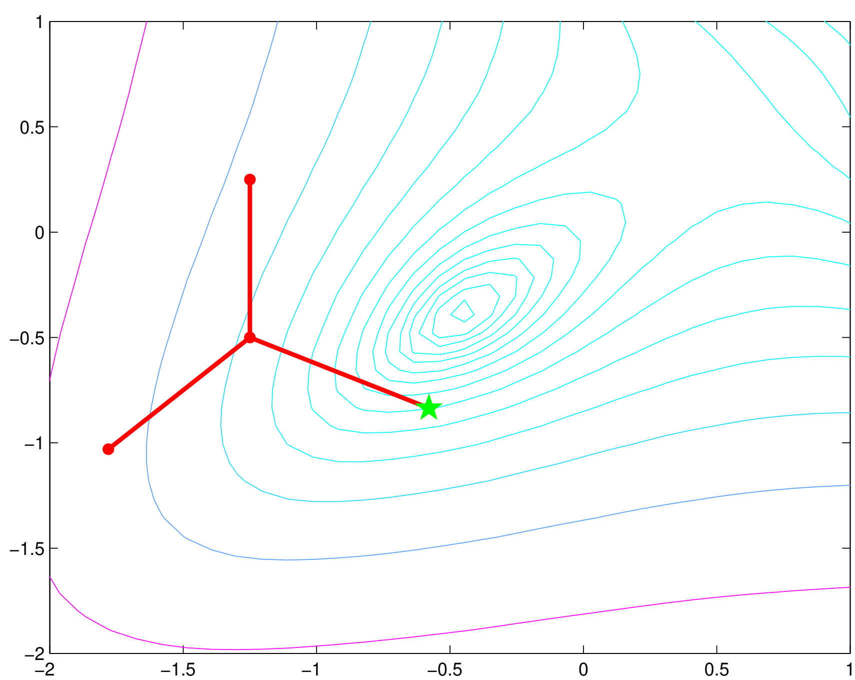
In *Direct Search Methods: Once Scorned, Now Respectable* (1996), Margaret Wright describes direct search methods as follows.

- A direct search uses only function values.
- A direct search method does not "in its heart" develop an approximate gradient.

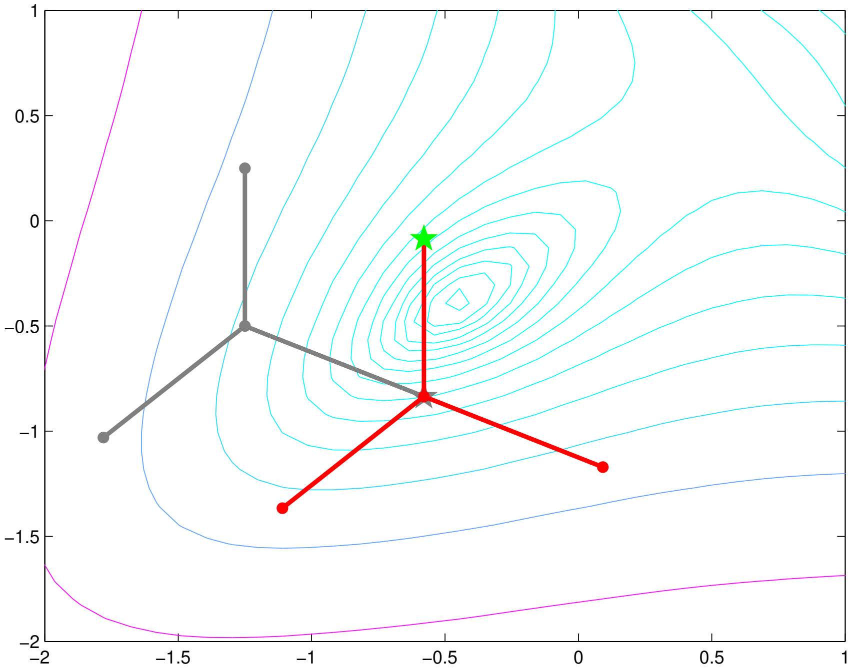
In particular, we are interested in *pattern search methods*, a type of direct search method.

- Popular in 1960s — Box, Hooke & Jeeves, etc.
- Multidimensional Search (MDS), Dennis & Torczon, 1991

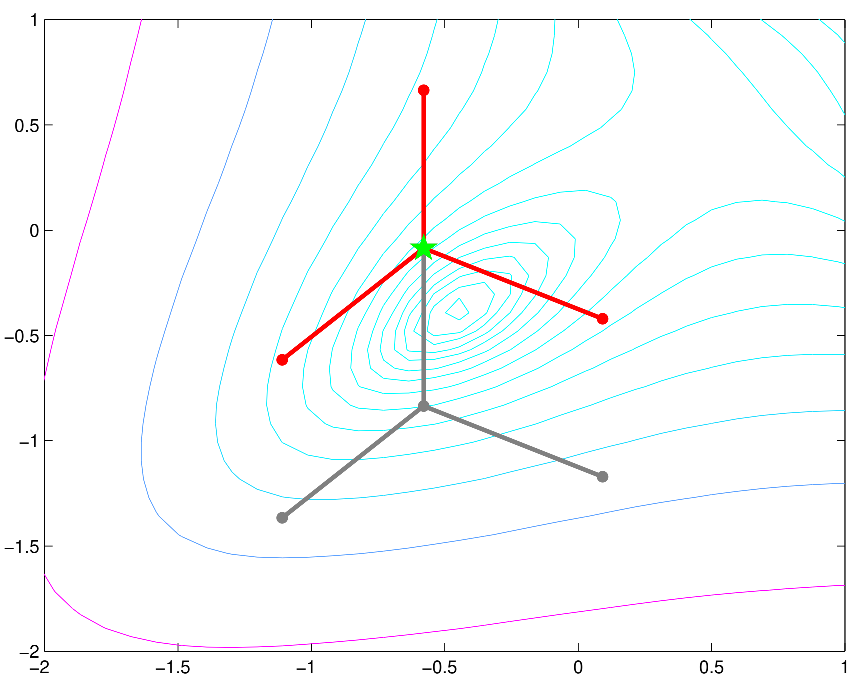
Pattern Search



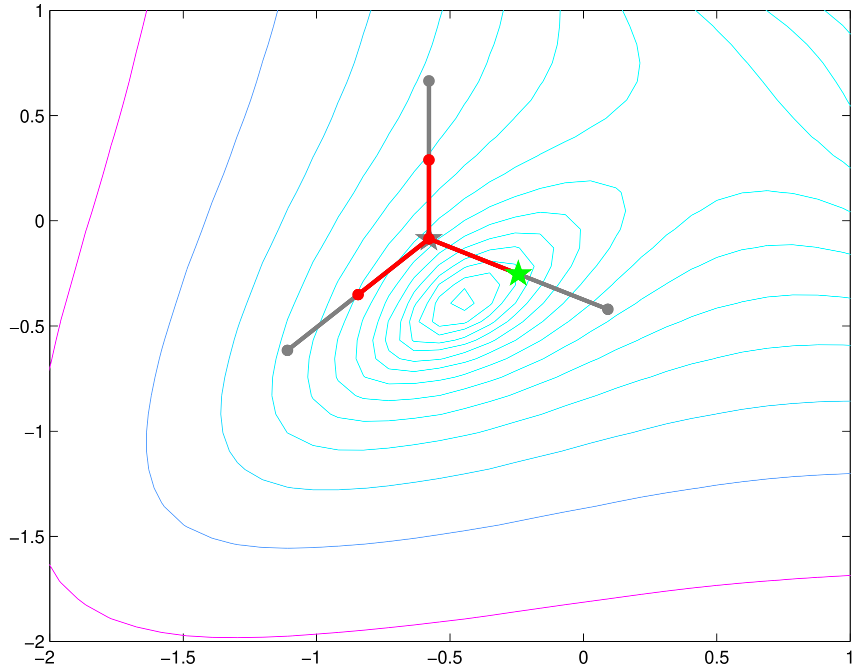
Pattern Search



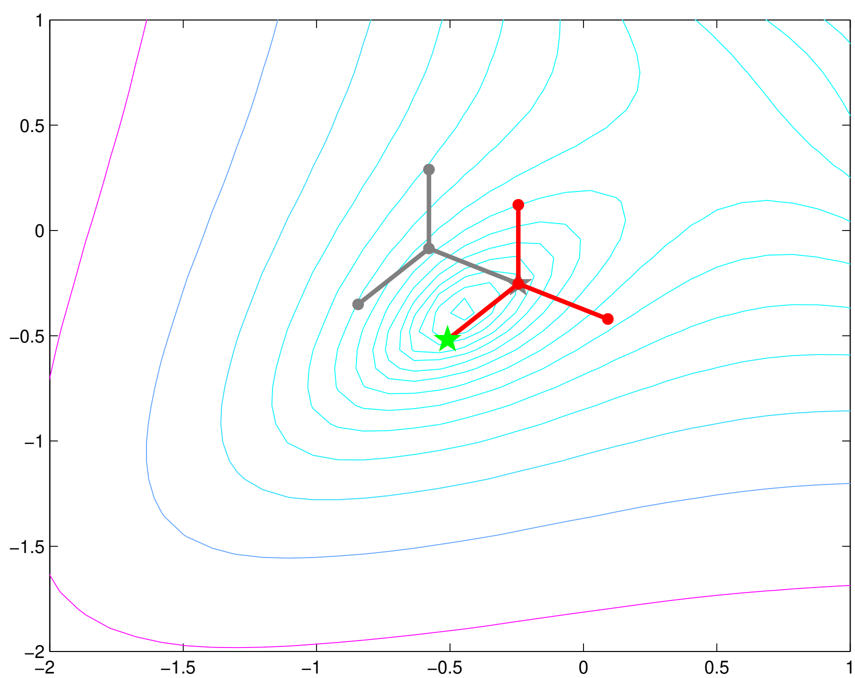
Pattern Search



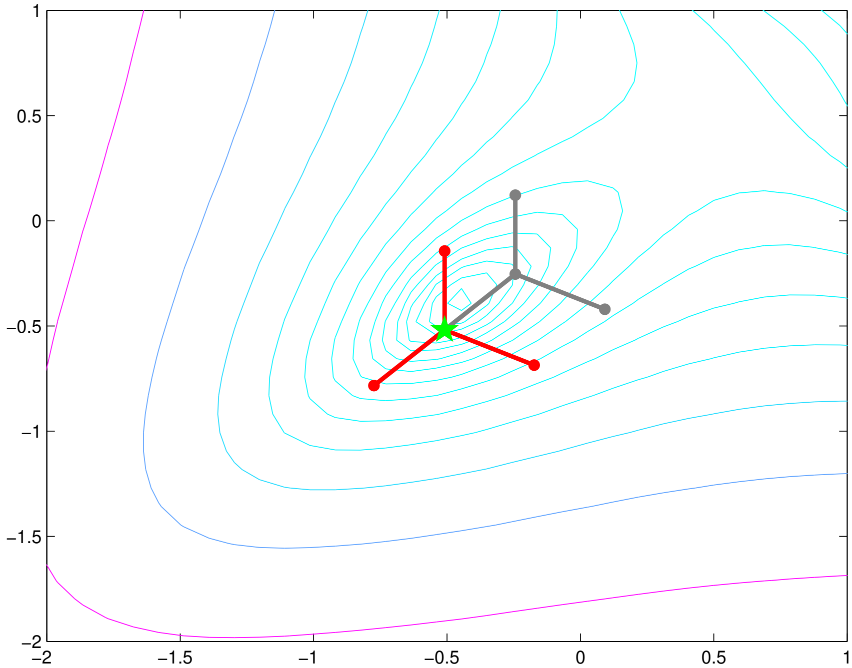
Pattern Search



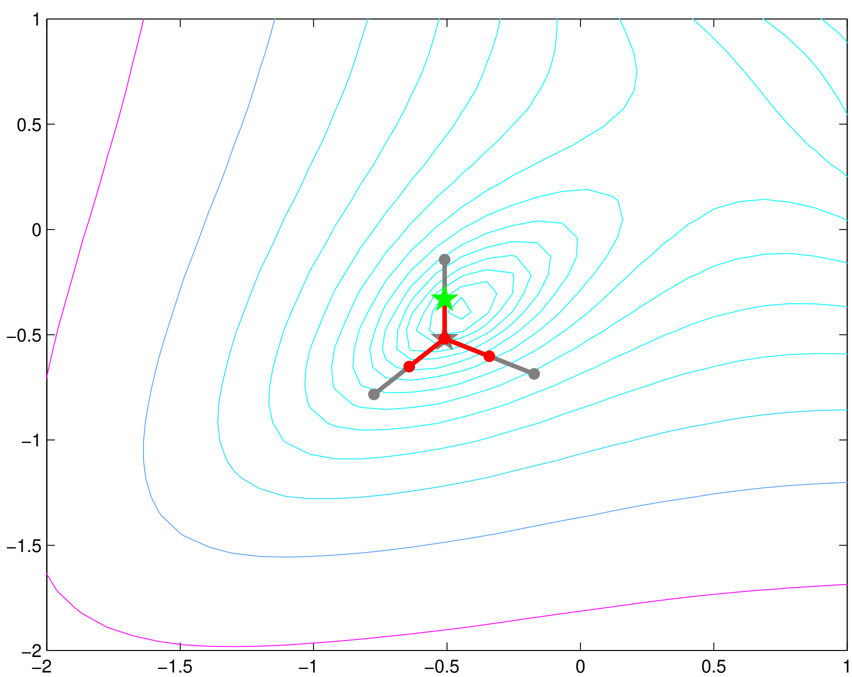
Pattern Search



Pattern Search



Pattern Search

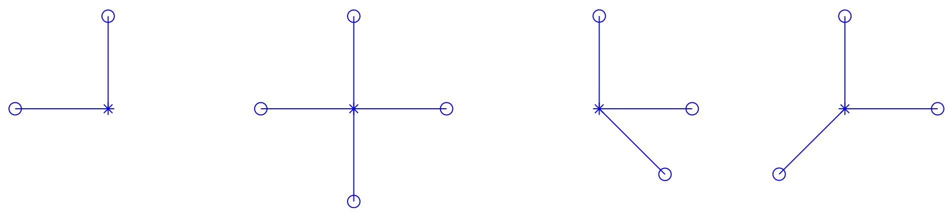


Positive Spanning Set

Defn: A set of vectors $\{d_1, \ldots, d_p\}$ positively spans $\Re^n$ if any vector $v \in \Re^n$ can be written as

$$v = \alpha_1 d_1 + \cdots + \alpha_p d_p, \quad \alpha_i \geq 0 \quad \forall\, i.$$

That is, any vector can be written as a *positive* linear combination of the vectors.



Observation: (Lewis & Torczon, 1996) If $\{d_1, \ldots, d_p\}$ positively spans $\Re^n$, then for any $g \neq 0$, there exists d_i such that $-g^T d_i > 0$.

On the Interaction Between Mathematicians & Computer Scientists

"On the typical university campus, the typical mathematician and the typical computing scientist live in different worlds: they don't know each other or, if they do, they are not on speaking terms."

— E. W. Dijkstra, 1986,
On A Cultural Gap,
The Mathematical Intelligencer.

Fields in Common to Computer Science & Math

- Numerical Analysis
- Linear Algebra
- Matrix Theory
- Optimization
- Graph Theory
- Algorithms
- Visualization
- Number Theory
- Cryptography
- Logic
- Set Theory
- Probability
- Combinatorics
- Etc…

How Math Benefits from Computer Science

The Four Color Theorem
Conjectured in 1852.
Proof found in 1879.
Proof wrong in 1890.
Proof, by computer, in 1976.

`http://www.math.gatech.edu/~thomas/FC/fourcolor.html`

"Computers are important, but not to mathematics."
— Paul Halmos (b. 1916)

Parallel Computing

"In pioneer days they used oxen for heavy pulling, and when one ox couldn't budge a log, they didn't try to grow a larger ox. We shouldn't be trying for bigger computers, but for more systems of computers."

—Grace Hopper, 1906–1992

Parallel Computing

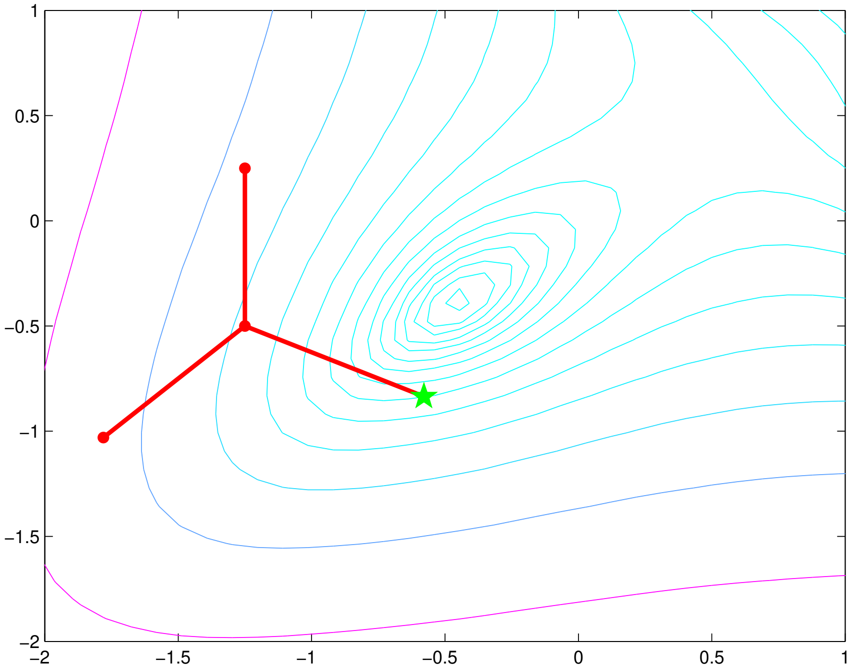
The challenge in parallel computing is rethinking our algorithms. Ideally, we could use p computers to perform a task p times faster, but we are constrained in a way described by *Amdahl's Law*.

$$T_1 \equiv T_s + (T_1 - T_s), \quad T_p = T_s + \frac{T_1 - T_s}{p}$$

$$\text{Speed-Up:} \quad \frac{T_1}{T_p} = \frac{T_1}{\left(T_s + \frac{T_1 - T_s}{p}\right)} \leq \frac{T_1}{T_s}$$

Redesigning algorithms for parallel computation can even lead to better serial algorithms — e.g., Jacobi-Davidson method for eigenvalues.

Parallel Pattern Search



Parallel Pattern Search Algorithm

This algorithm is from a **single** *processor's point-of-view.*

1. Compute $x_{\text{trial}} \leftarrow x_{\text{best}} + \Delta_{\text{trial}}\, d$, and evaluate $f_{\text{trial}} \leftarrow f(x_{\text{trial}})$.
2. Perform a *global reduction* to determine f_+ (and the associated x_+) such that f_+ is the minimum of all f_{trial} values on all processors.
3. (a) If $f_+ < f_{\text{best}}$, then $\{x_{\text{best}}, f_{\text{best}}\} \leftarrow \{x_+, f_+\}$.
   (b) Else $\Delta_{\text{trial}} \leftarrow \frac{1}{2}\Delta_{\text{trial}}$.
4. If $\Delta_{\text{trial}} > tol$, go to Step 1. Else, exit.

New Issues

- Simulations are more complicated, and the execution time may vary substantially depending on the inputs. Further, simulations may takes minutes, hours, or evens days to execute.
- The parallel platform is now a heterogeneous cluster of workstations rather than a tightly coupled network of homogeneous processors.
- Faults — both of the simulation and the hardware — are real issues.

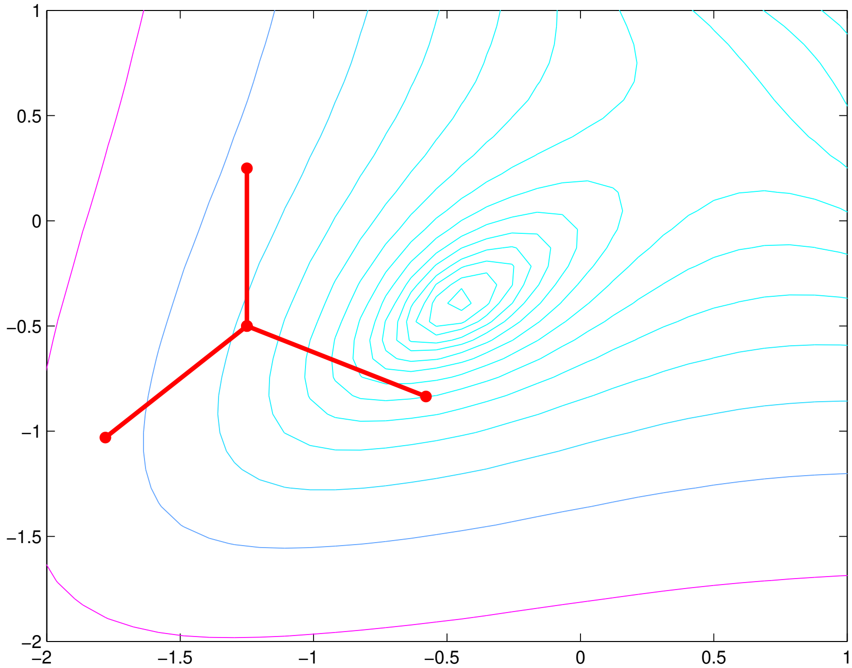
We are motivated to revisit parallel pattern search
to take these new issues into account...

Asynchronous Parallel Pattern Search (APPS)

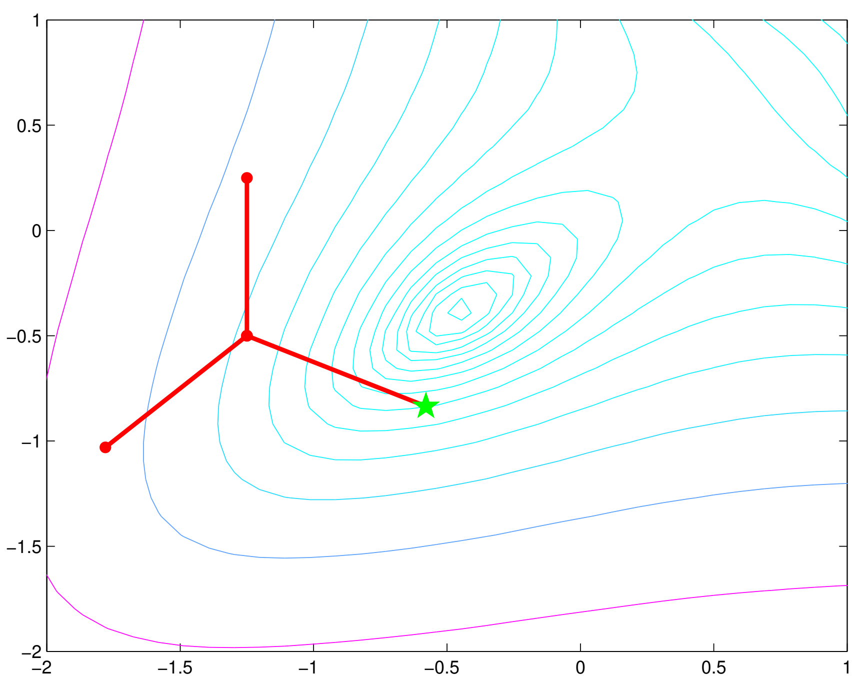
"In great mathematics there is a very high degree of unexpectedness, combined with inevitability and economy."

— Godfrey Hardy (1877 - 1947)
A Mathematician's Apology, 1941.

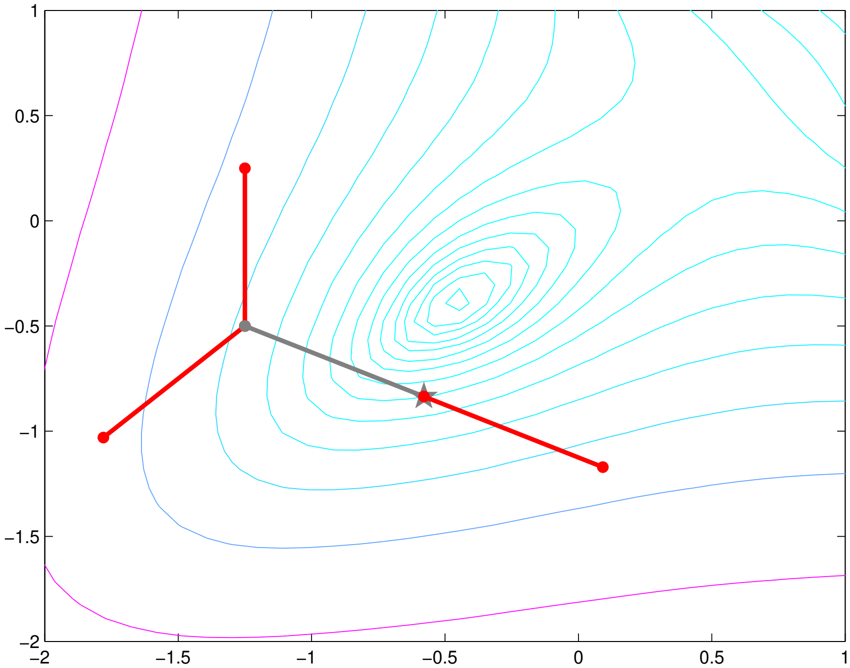
Asynchronous Parallel Pattern Search



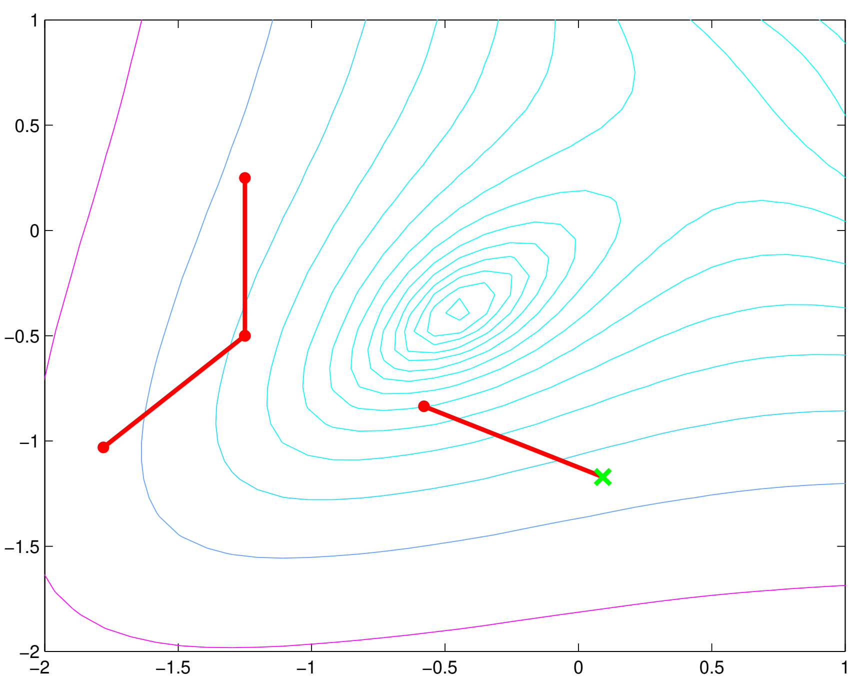
Asynchronous Parallel Pattern Search



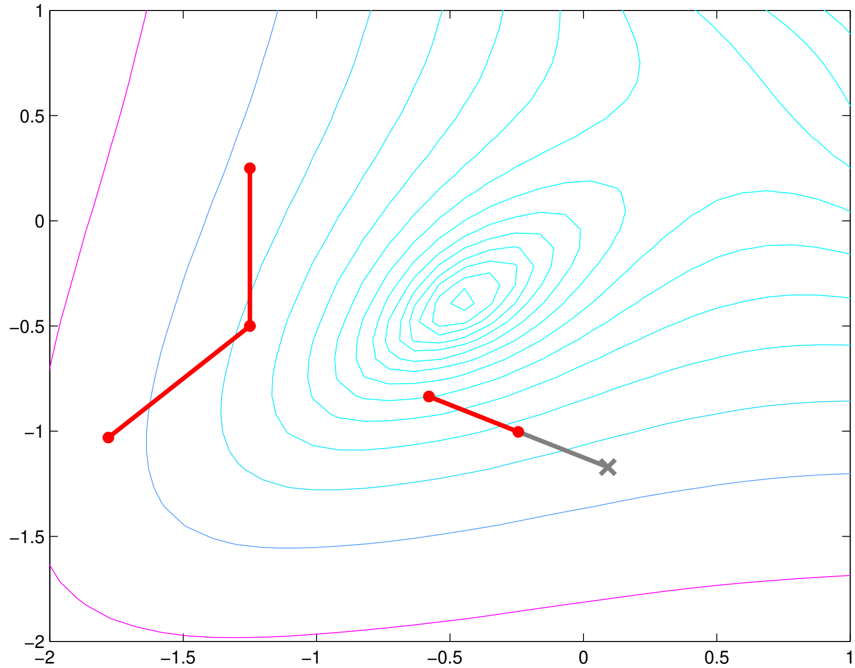
Asynchronous Parallel Pattern Search



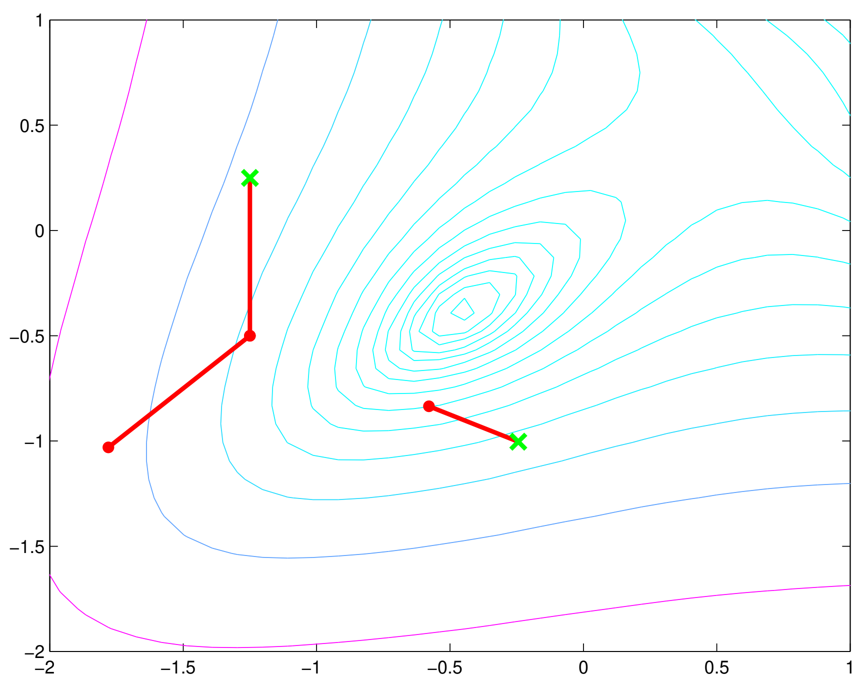
Asynchronous Parallel Pattern Search



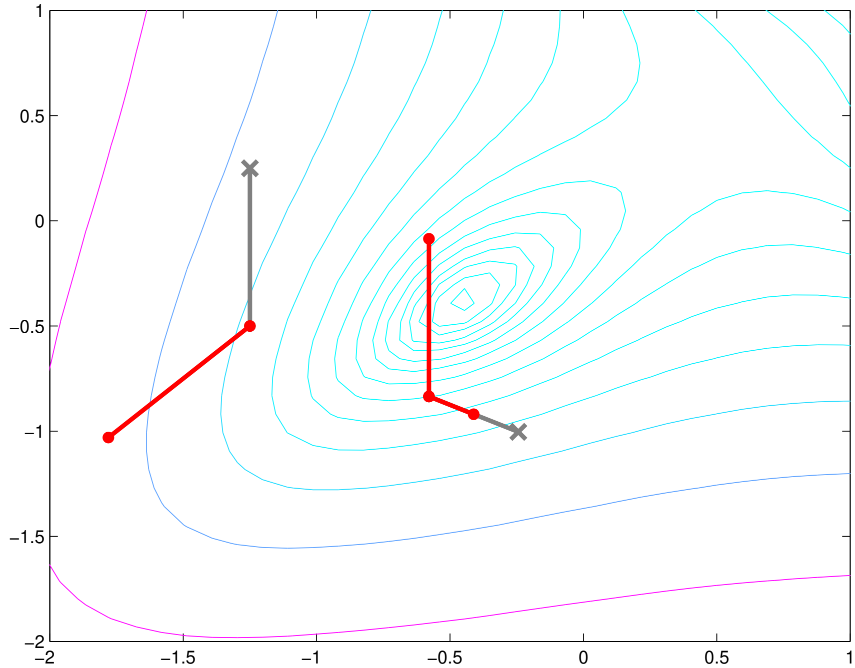
Asynchronous Parallel Pattern Search



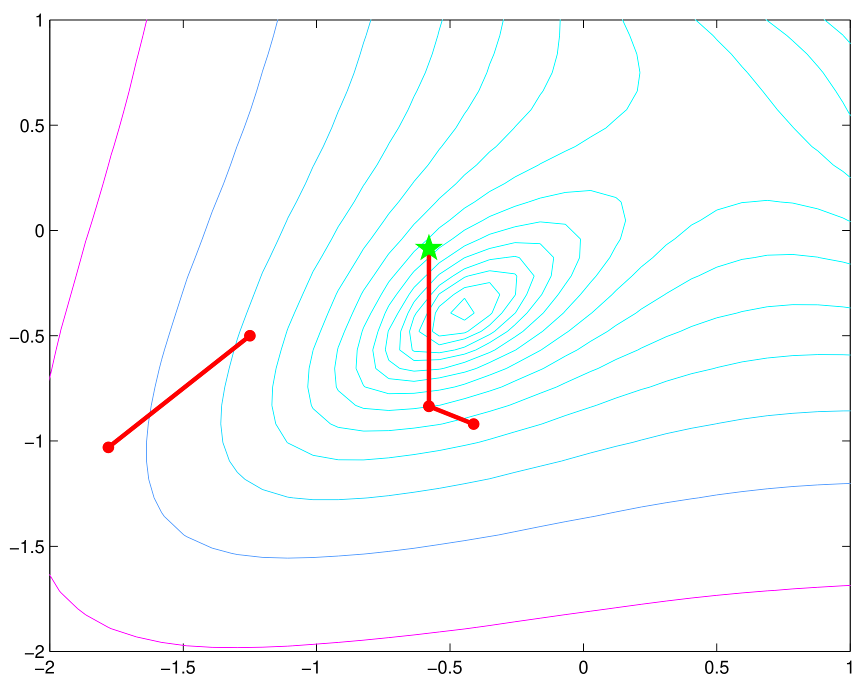
Asynchronous Parallel Pattern Search



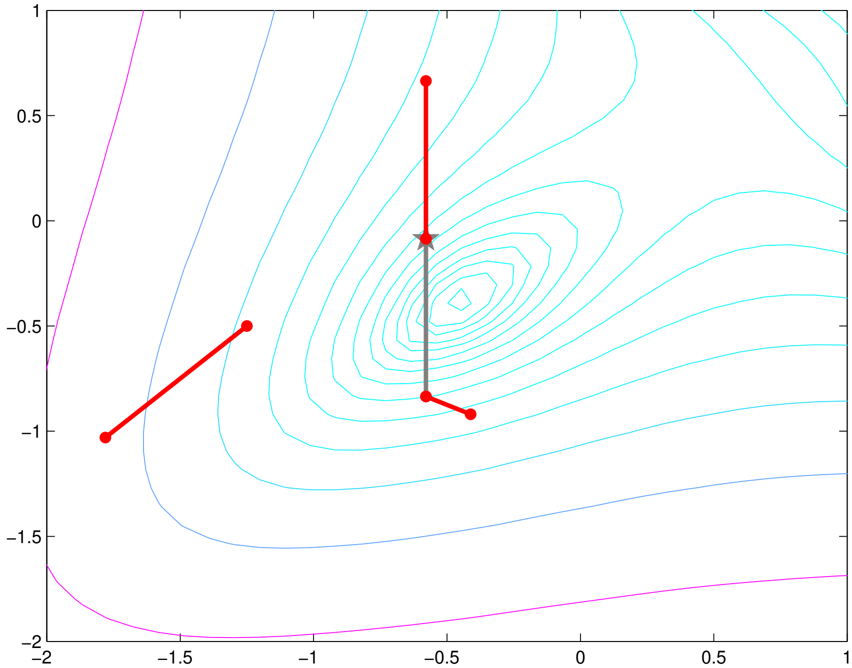
Asynchronous Parallel Pattern Search



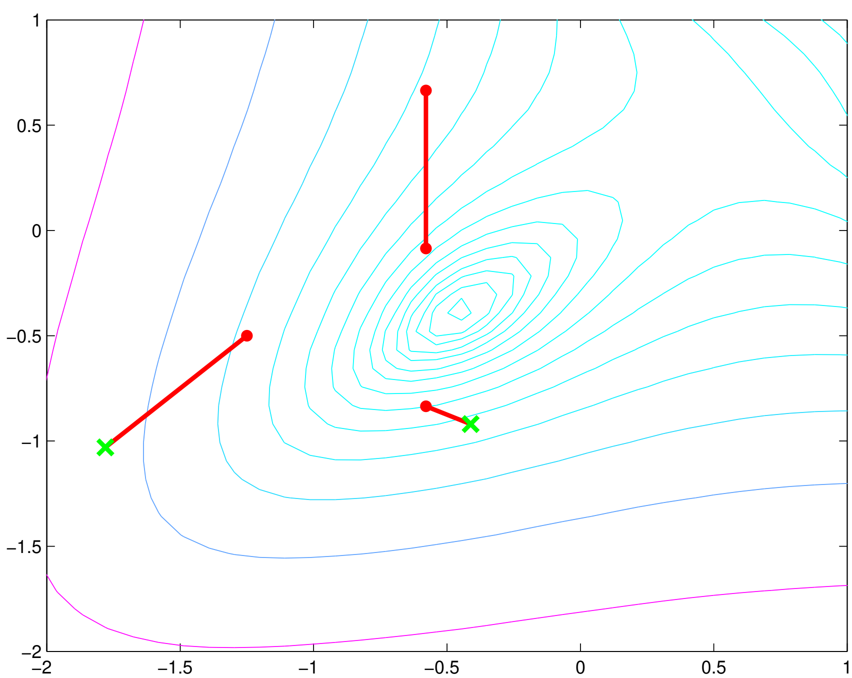
Asynchronous Parallel Pattern Search



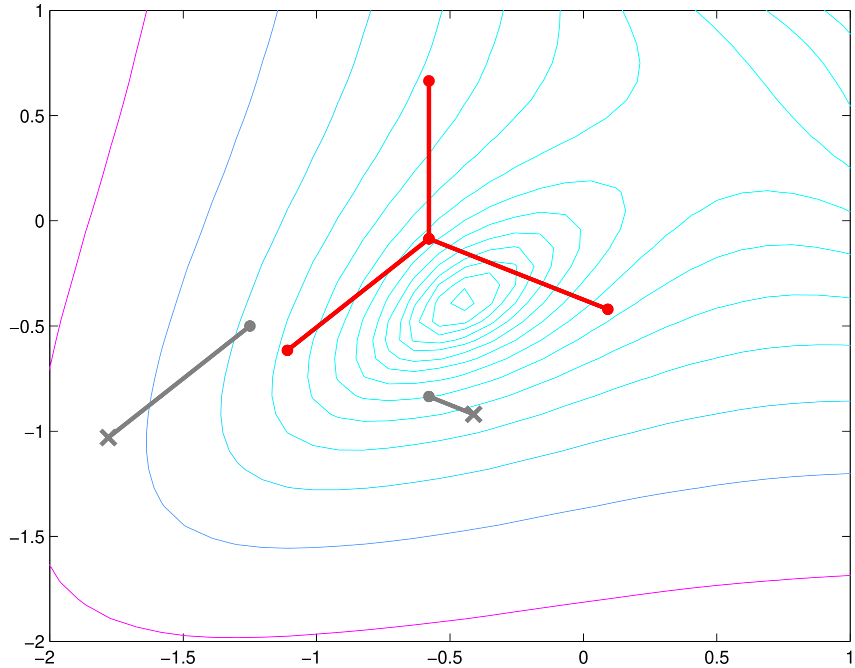
Asynchronous Parallel Pattern Search



Asynchronous Parallel Pattern Search



Asynchronous Parallel Pattern Search



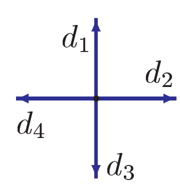
0. For each **new best** message in my queue:
   (a) If $f_+ < f_{\text{best}}$ for an incoming triplet $\{x_+, f_+, \Delta_+\}$, then $\{x_{\text{best}}, f_{\text{best}}, \Delta_{\text{best}}\} \leftarrow \{x_+, f_+, \Delta_+\}$, and $\Delta_{\text{trial}} \leftarrow \Delta_{\text{best}}$.
   (b) Else, discard the triplet $\{x_+, f_+, \Delta_+\}$.
1. Compute $x_{\text{trial}} \leftarrow x_{\text{best}} + \Delta_{\text{trial}}\, d$ and evaluate $f_{\text{trial}} \equiv f(x_{\text{trial}})$.
2. Set $\{x_+, f_+, \Delta_+\} \leftarrow \{x_{\text{trial}}, f_{\text{trial}}, \Delta_{\text{trial}}\}$.
3. (a) If $f_+ < f_{\text{best}}$, then $\{x_{\text{best}}, f_{\text{best}}, \Delta_{\text{best}}\} \leftarrow \{x_+, f_+, \Delta_+\}$, and *broadcast* a **new best** message with $\{x_{\text{best}}, f_{\text{best}}, \Delta_{\text{best}}\}$.
   (b) Else $\Delta_{\text{trial}} \leftarrow \frac{1}{2}\Delta_{\text{trial}}$.
4. If $\Delta_{\text{trial}} > tol$, go to Step 0. Else *broadcast* a **single direction convergence** message with the triplet $\{x_{\text{best}}, f_{\text{best}}, \Delta_{\text{best}}\}$.
5. Wait and process each incoming message in my queue until either
   (a) enough of the processes report single direction convergence for this same point or (b) a better point is received. In case

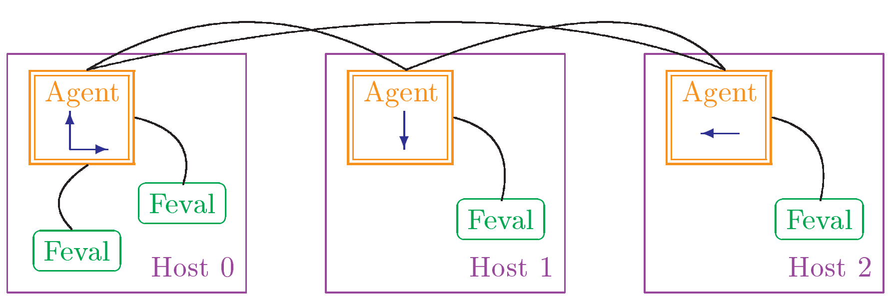
(a), exit. In case (b), go to Step 0.

APPS Implementation

- The work of the pattern search is divided among *agents*, one per host.
- Each agent is in charge of some subset of the *search pattern*.
- Agents make algorithmic decisions based only on available information — *there is no synchronization*.

2-D Search Pattern

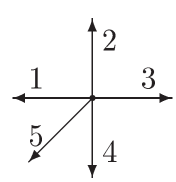




Play APPS Movie

Checking Convergence

2-D Search Pattern



We have *convergence* when enough processes have locally converged to contain a positive basis.

To verify that a given set $\mathcal{V} = \{v_1, v_2, \ldots, v_m\}$ is a positive spanning set, solve $n+1$ nonnegative least squares problems of the form

$$\min \| Vx - b \| \text{ s.t. } x_i \geq 0 \; \forall i$$

for $b = \{e_1, \ldots, e_n, -e\}$. All have residual zero iff V is a positive spanning set.

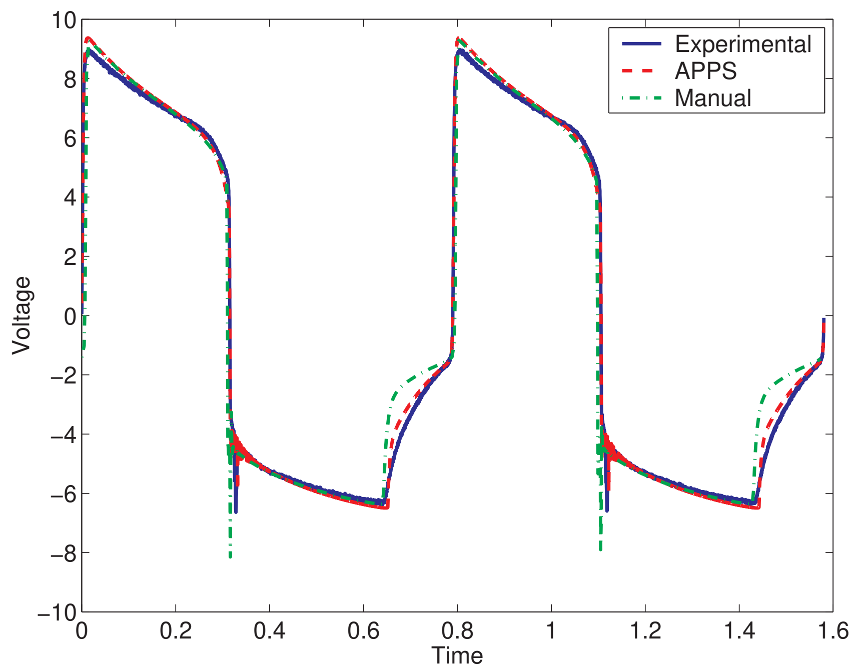
Circuit Simulation Problem Numerical Results

$$f(x) = \sum_{t=1}^{N} \left(V_t^{\mathrm{SIM}}(x) - V_t^{\mathrm{EXP}} \right)^2 \quad \text{(17 variables)}$$

- Search directions are $\pm$ Unit Vectors (34) plus Random
- The variables are bound constrained; i.e., $l_i \leq x_i \leq u_i$

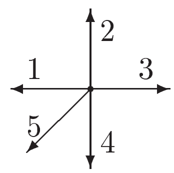
Method	Procs	f(x∗)	Function Evals	Idle Time	Total Time
APPS	34	26.2	57.5	111.92	1330.55
APPS	50	26.9	50.6	63.22	807.29
PPS	34	28.8	53.0	521.48	1712.24
PPS	50	34.9	47.0	905.48	1646.53

Circuit Simulation Problem Solution



Fault Tolerance in APPS

2-D Search Pattern



- We can afford to lose processes as long as we maintain a positive basis.
- Processes can be restarted when we no longer have a positive basis.
- Progression towards a solution continues as long as one or more APPS processes is alive.
- Completion does *not* depend on the survival of any particular APPS agent.

Play APPS with Faults Movie

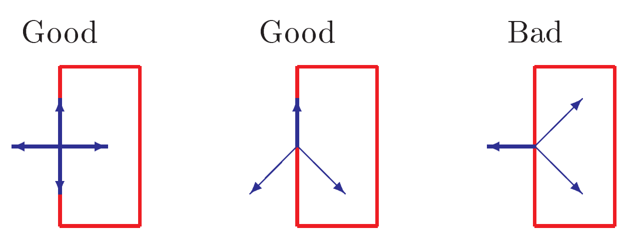
Circuit Simulation with Faults

The "fault" versions have a failure in the function evaluation or agent every 30 seconds.

Method	**Initial Procs**	**f(x∗)**	**Total Time**
APPS	34	26.2	1330.55
APPS	34-faults	27.8	1618.46
PPS	34	28.8	1712.25
APPS	50	26.9	807.29
APPS	50-faults	54.2	1041.14
PPS	50	34.9	1646.53

APPS with Bound Constraints

- Must use plus/minus unit vectors for search.
- Infeasible points are mapped to the boundary.
- Change the positive basis test for convergence.
- Restarting procedure for failed search directions must always keep *base pattern* active.



Checking Convergence for Bound Constrained APPS

We have convergence when enough *projected* search directions have converged to contain a *projected* positive basis.

Given a set of vectors $\mathcal{V} = \{v_1, v_2, \ldots, v_m\}$, we verify that $\mathcal{V}$ is a *projected* positive spanning set by solving a series of NNLS problems of the form

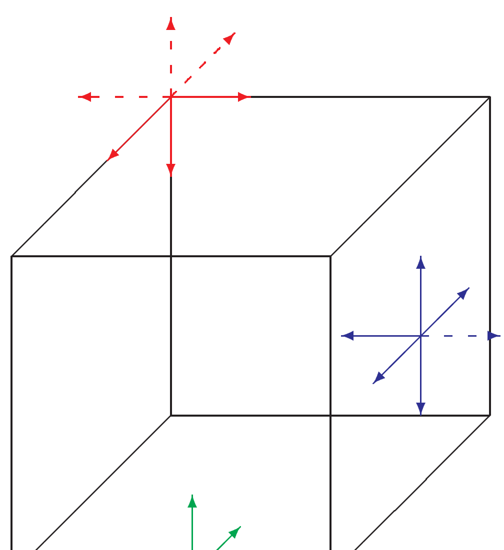
$$\| \, P(V)x - b \, \| \;\; \text{s.t. } x_i > 0 \; \forall i$$

for each $b \in \mathcal{B}$, where

$$\begin{aligned} \mathcal{B} \;\; = \;\; & \{ \; e_i \mid \text{upper bound } i \text{ is inactive } \; \} \cup \\ & \{ -e_i \mid \text{lower bound } i \text{ is inactive } \} \end{aligned}$$

All have residual zero iff V is a positive spanning set.

Active & Inactive Bound Constraints



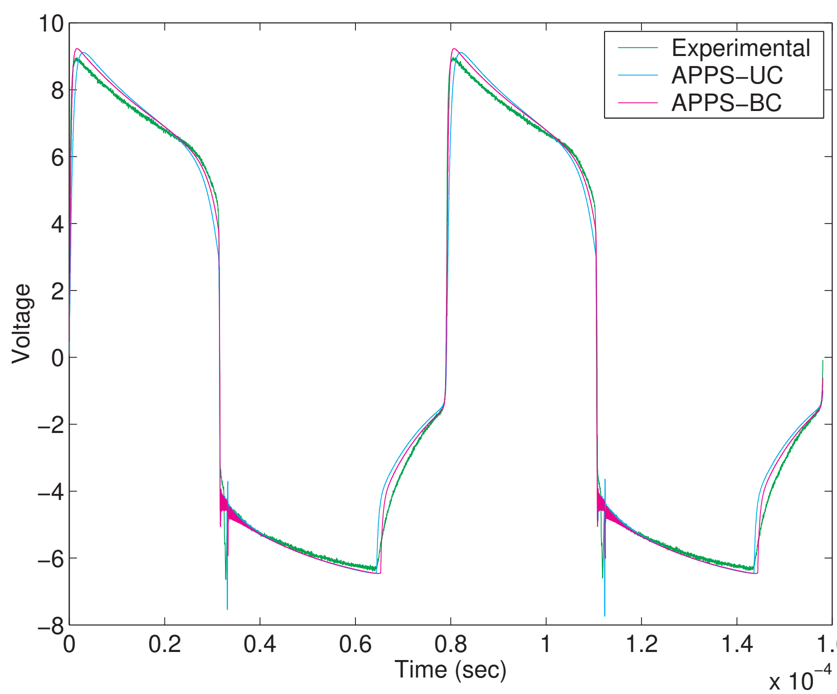
Numerical Results for Circuit Problem

Type	Search Dirs	Final f(x)	Time (sec)	Idle (sec)	Func. Evals	No-Gos
BC	34	25.3	2750	448	36	3
UC	34	45.1	2440	414	39	4

BC = Bound Constrained

UC = Unconstrained

Circuit Problem Results



APPSPACK

The software implementation of the APPS algorithm.

- PVM or MPI
- Bound constrained or unconstrained
- Function value cache
- Surrogate models

Freely available under the GNU L-GPL license.

What I haven't talked about...

- The Gory Details
- Convergence Theory
- Implementation
- Surrogate Models
- Linear & Nonlinear Constraints
- Etc...

"The most valuable acquisitions in a scientific or technical education are the general-purpose mental tools which remain serviceable for a lifetime. I rate natural language and mathematics as the most important of these tools, and computer science as a third."

— George Forsythe, 1968,
What to do till the computer scientist comes home,
American Mathematical Monthly.

Thanks To...

"It is amazing what you can accomplish if you do not care who gets the credit." — Harry Truman

- Virginia Torczon, Wm & Mary
- Patty Hough, Sandia
- Alton Patrick, Summer Intern
- Sarah Brown, Summer Intern
- Ken Marx, Sandia Engineer

For More Information...

http://csmr.ca.sandia.gov/projects/apps.html
http://csmr.ca.sandia.gov/~tgkolda/
tgkolda@sandia.gov