STORM: Stochastic Optimization Using Random Models

Katya Scheinberg

Lehigh University

(Joint work with R. Chen and M. Menickelly)

Outline

- Stochastic optimization problem
  - black box
  - gradient based
- Existing methods vs. this work
- Algorithm, assumptions and analysis
- Computational results.

Black-box stochastic optimization

- Unconstrained optimization problem

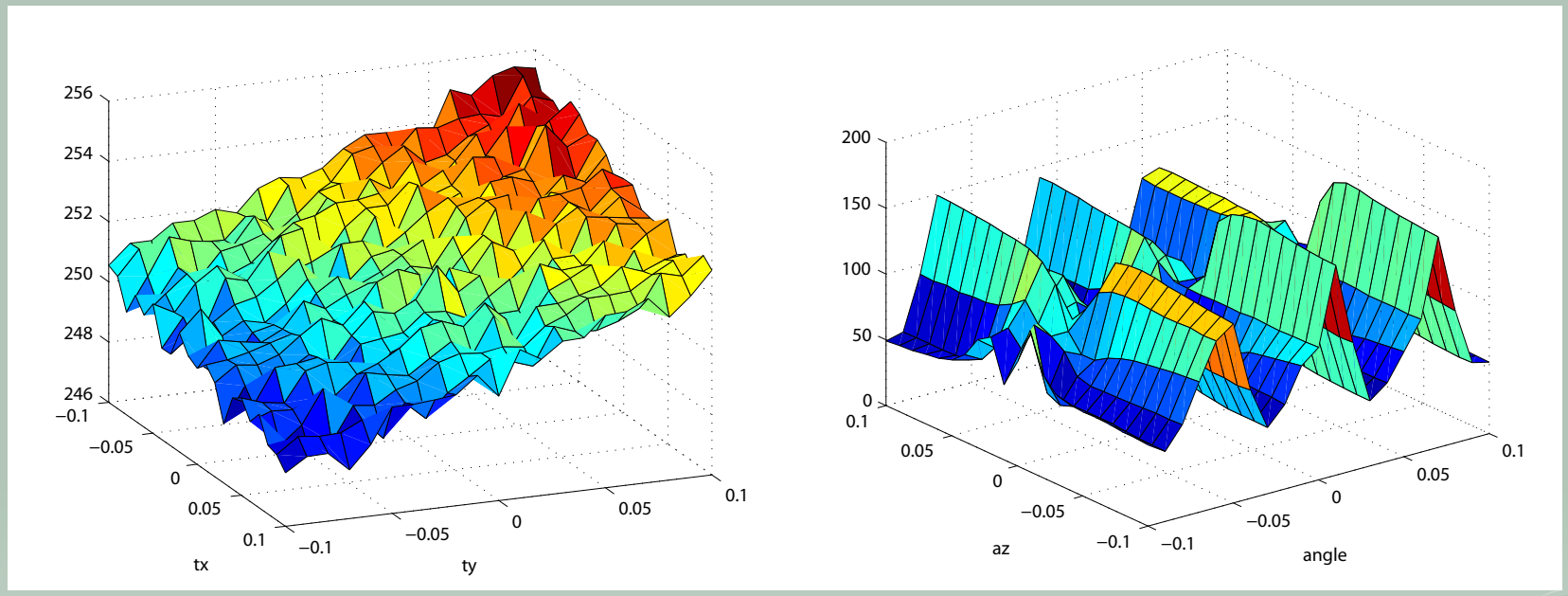
$$\min_{x \in \Omega} f(x)$$

- Function $f \in C^1$ or C^2 and bounded from below.
- $f(x)$ cannot be computed, instead..

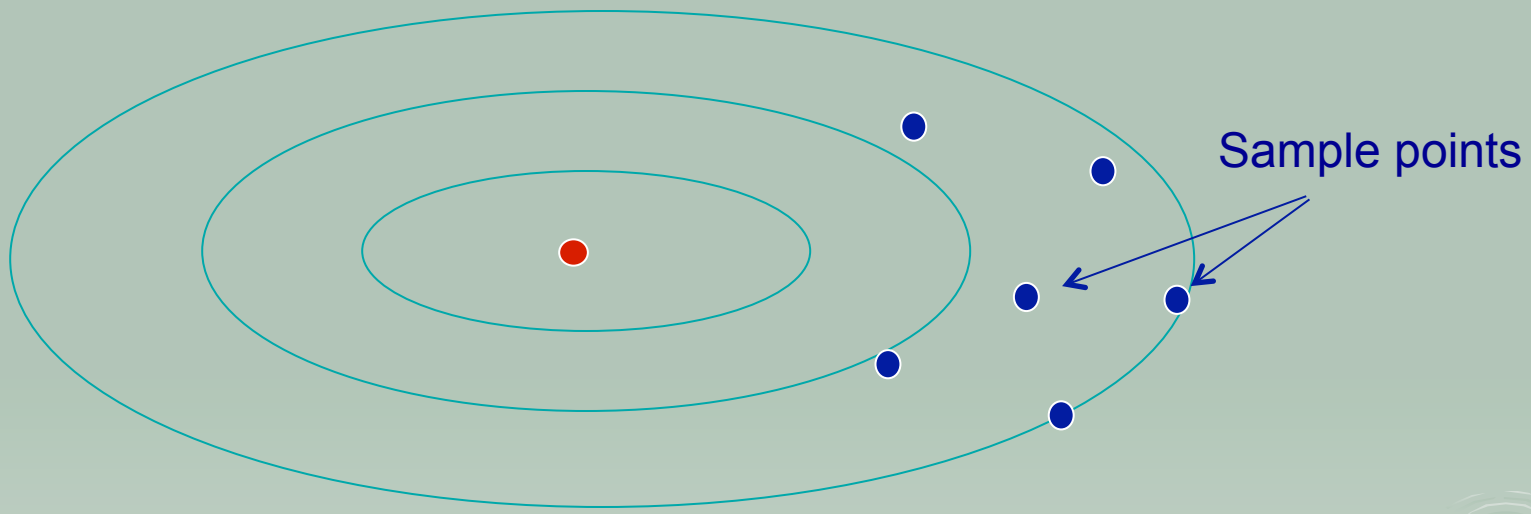
$$\tilde{f}(x) = f(x, \varepsilon)$$

- where ϵ *is a random variable*
- If $E_\epsilon[f(x,\epsilon)]=f(x)$, then the noise *is unbiased*
- If $E_\epsilon[f(x,\epsilon)]=h(x) \neq f(x)$, then the noise *is biased*

Noisy computable function



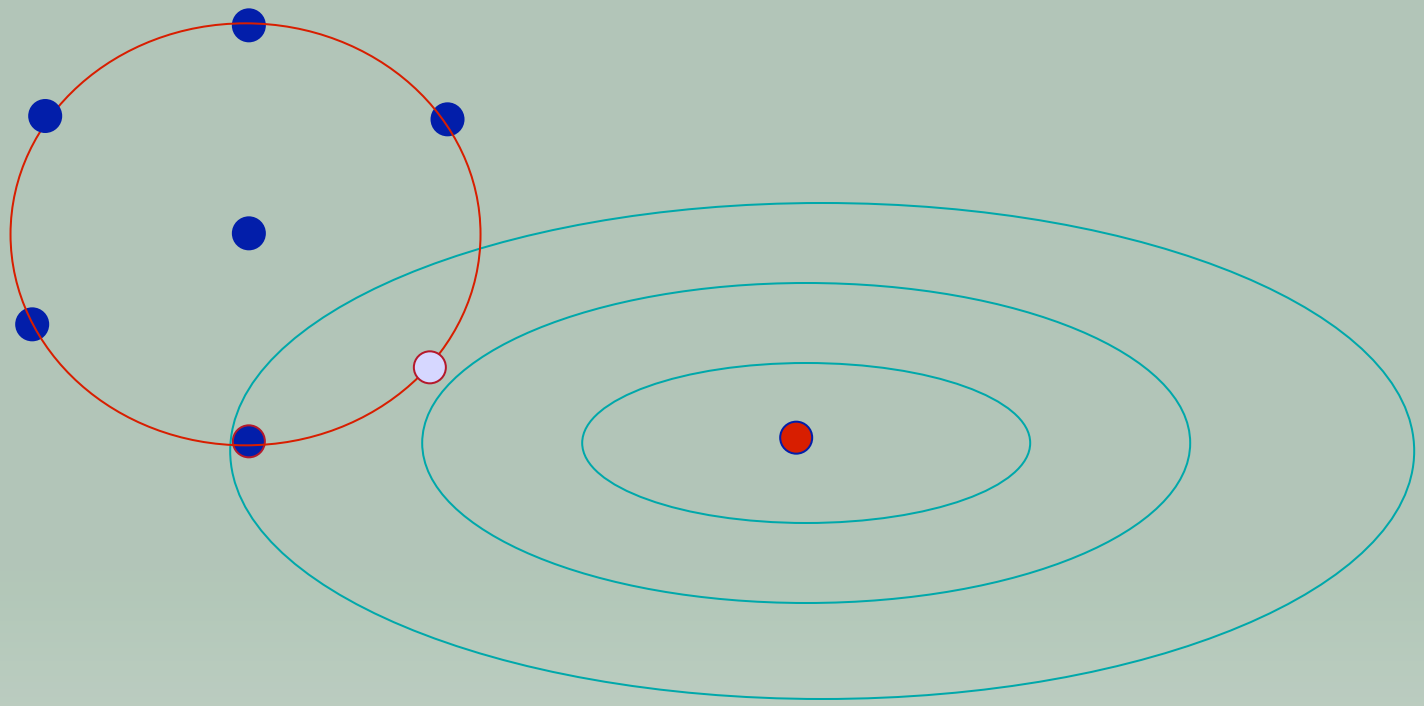
Sampling the black box function



How to choose and to use the sample points and the functions values defines different methods

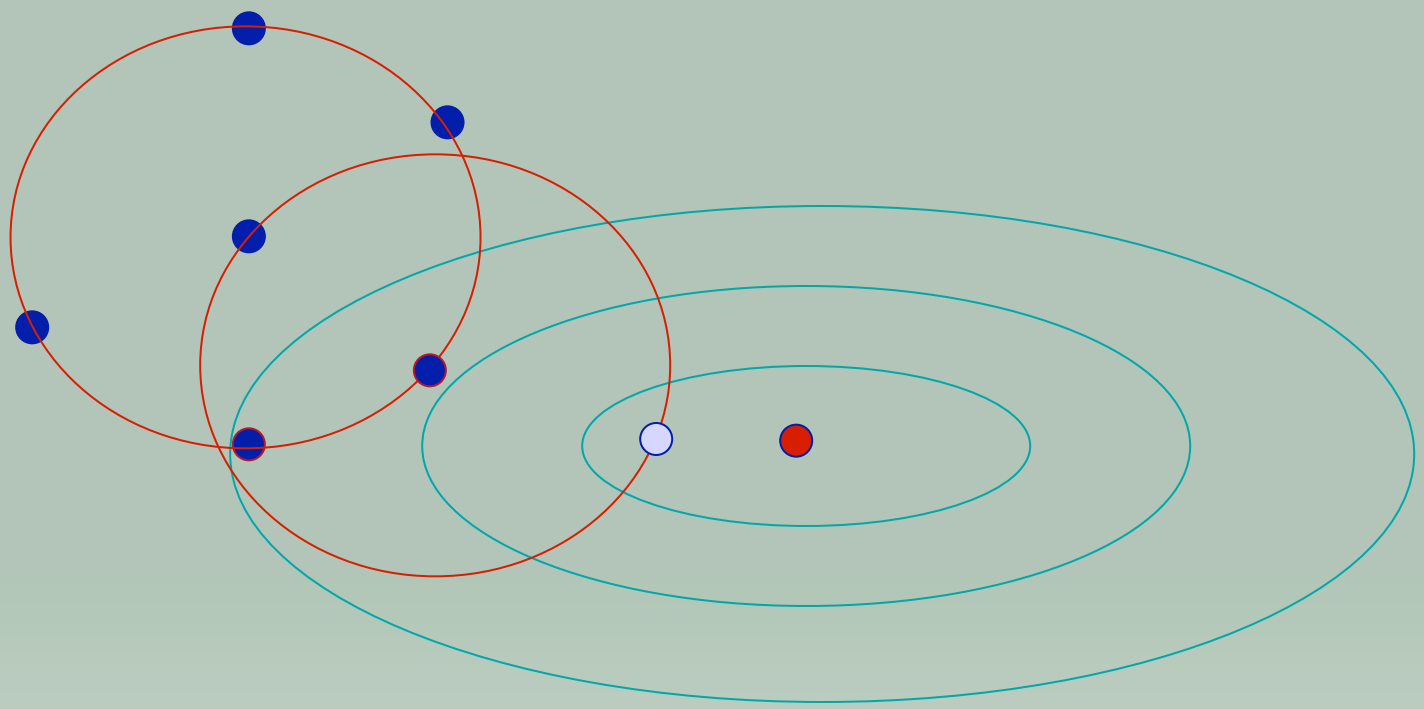
See book by Conn, S. and Vicente, 2009

Model based trust region methods



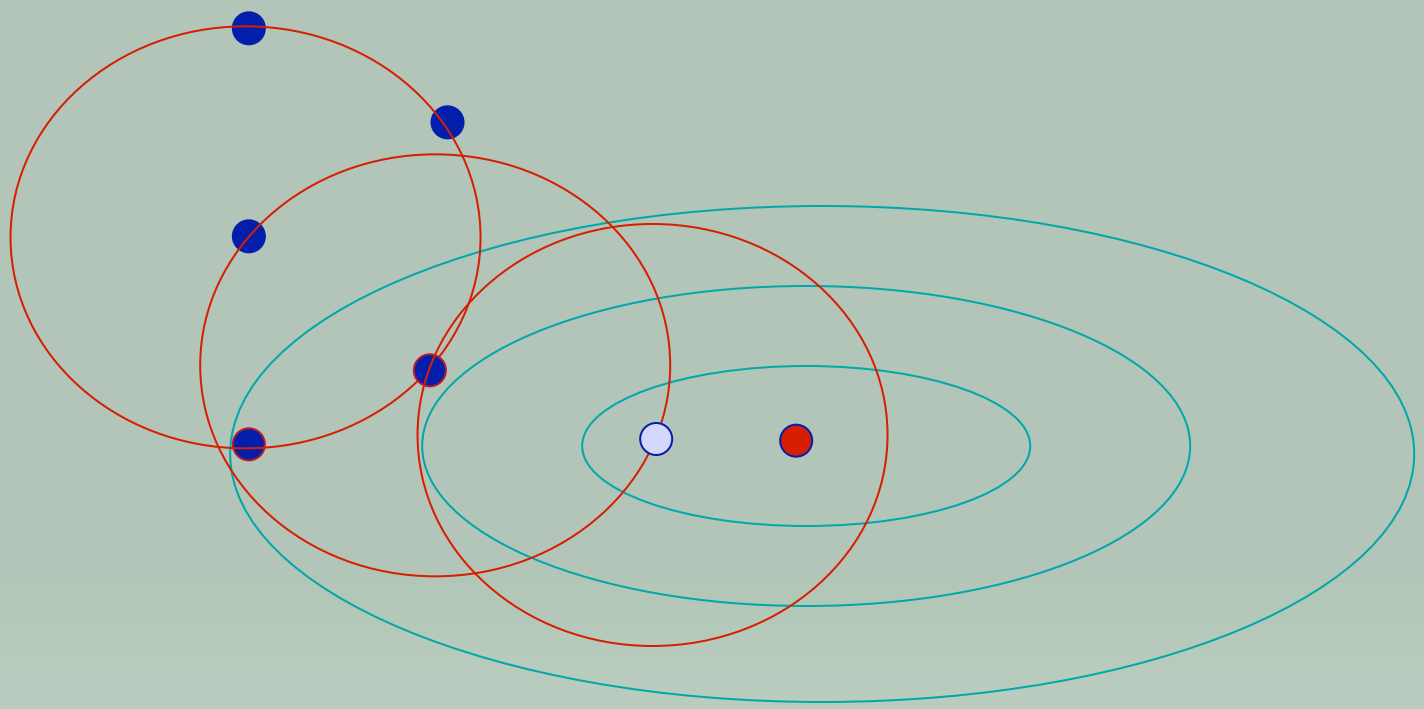
Powell, Conn, S. Toint, Vicente, Wild, etc.

Model based trust region methods



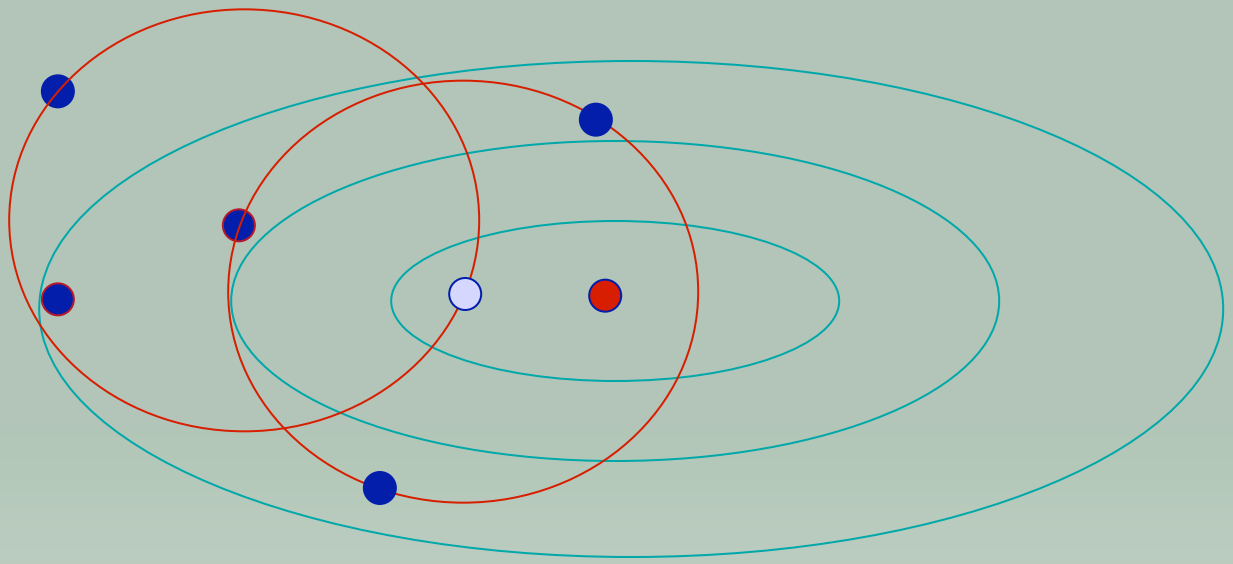
Powell, Conn, S. Toint, Vicente, Wild, etc.

Model based trust region methods



Powell, Conn, S. Toint, Vicente, Wild, etc.

Model Based trust region methods



Exploits curvature, flexible efficient steps, uses second order models.

Gradient-based stochastic optimization

- Unconstrained optimization problem

$$\min_{x \in \Omega} f(x)$$

- Function $f \in C^1$ or C^2 and bounded from below.
- $f(x)$ or $\nabla f(x)$ cannot be computed, instead.

$$\tilde{f}(x) = f(x, \varepsilon) \quad \tilde{\nabla} f(x) = g(x, \varepsilon)$$

where ϵ *is a random variable*

- If $E_\epsilon[g(x,\epsilon)] = \nabla f(x)$, then the noise *is unbiased*
- If $E_\epsilon[g(x,\epsilon)] = h(x) \neq \nabla f(x)$, then the noise *is biased*

What methods exist for stochastic optimization?

- Stochastic gradient
- Sample average (simulation optimization)
- Sample path optimization
- Methods based on random models (ours).

Stochastic gradient descent

(Robbins-Monro, 51, Polyak-Yuditski, 92, Spall'00, Shalev-Shwartz,11, Ghadimi, Lan 13)

- Assume unbiased estimator of the gradient can be computed

$$\tilde{\nabla} f(x) = g(x, \varepsilon) \qquad \nabla f(x) = E_{\varepsilon}[g(x, \varepsilon)]$$

- The method then is:

$$x^{k+1} = x^k - \alpha^k \tilde{\nabla} f(x^k), \quad \alpha_k \to 0, \ \sum_{i=1}^{k} \alpha_k = \infty$$

- Many variants exits, but in most
  - tuning α_k sequence is required,
  - convergence is slow.

Sample averaging

(Shapiro, Homem-De-Mello, Pasupathy, Ghosh, Glynn, etcl)

- Assume unbiased estimator of the gradient can be computed

$$\tilde{\nabla} f(x) = g(x, \varepsilon) \qquad \nabla f(x) = E_{\varepsilon}[g(x, \varepsilon)]$$

- Compute a sample average gradient at x^k, given sample size S_k:

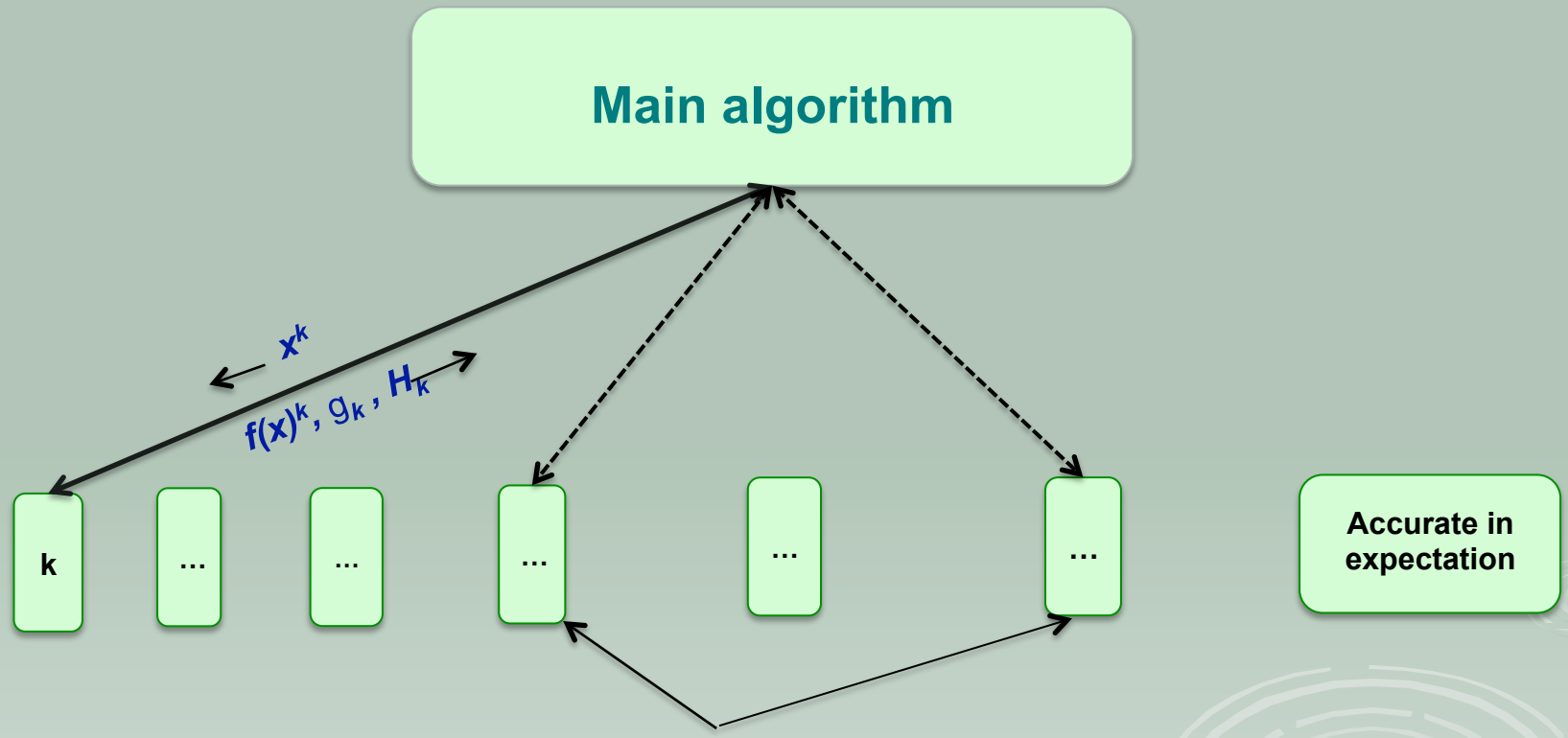
$$f_{S_k}(x^k) = \frac{1}{S_k} \sum_{i=1}^{S_k} f(x^k, \varepsilon_i) \quad \nabla_{S_k} f(x^k) = \frac{1}{S_k} \sum_{i=1}^{S_k} g(x^k, \varepsilon_i)$$

$$x^{k+1} = x^k - \alpha^k \nabla_{S_k} f(x^k), \quad \alpha_k \not\to 0,$$

- Tend to work well in practice, many variants exist, but strong assumptions needed in theory.

Stochastic gradient

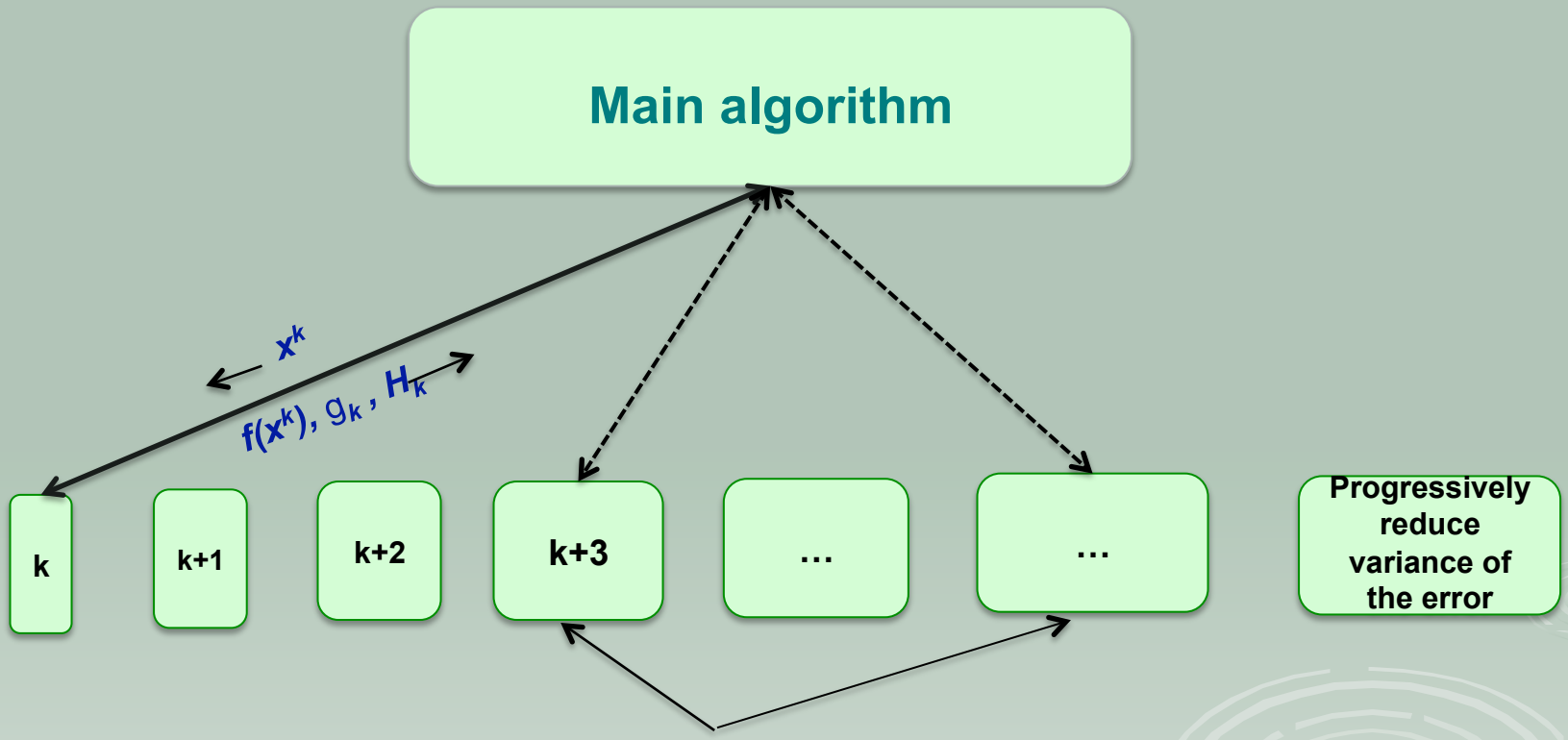
Accurate in expectation. Accuracy does not improve. Iterations remain inexpensive! Does not apply to standard frameworks. Convergence rates are usually lower.



Approximate computation of : $f(x^k),\ g_k,\ H_k$

sample average approximation

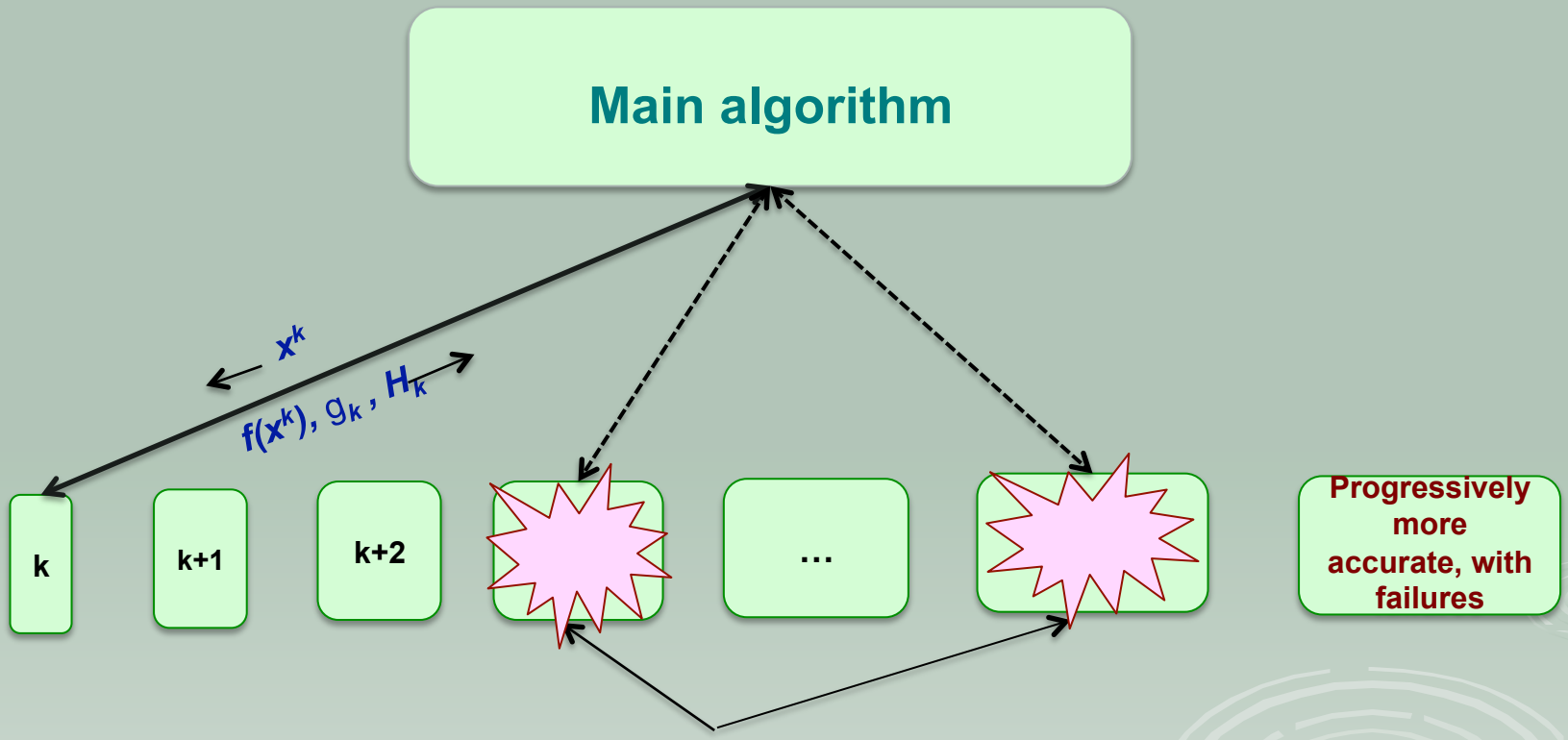
Information gets more and more accurate as needed.
This is usually achieved by resampling gradient and function
Information is assumed to be accurate in expectation with bounded variance.
Under sufficient sampling and unbiased noise assumptions preserves the convergence rates.



Approximate computation: $f(x^k),\ g_k,\ H_k$

Random inexact first (and second) order models.

Information gets more and more exact as needed.
But this only has to hold with some probability.
No assumption on distribution, or expectation.
Applies to standard deterministic frameworks.
Preserves the convergence rates.



Approximate computation: $f(x^k),\ g_k,\ H_k$

Biased and unbiased noise examples.

- Noisy function samples.
  - Unbiased nose $E(f(x,\varepsilon)) = f(x) \quad \forall x$

$$f(x,\varepsilon) = f(x) + \varepsilon, \text{or} f(x,\varepsilon) = f(x)(1+\varepsilon),\ E(\varepsilon) = 0$$

  - Biased noise

$$\tilde{f}(x) = f(x,\varepsilon) = \begin{cases} f(x), & \text{w.p. } p \\ \varepsilon(x) \leq V & \text{w.p. } 1-p, \end{cases}$$

$$E(f(x,\epsilon)) = pf(x) + (1-p)\varepsilon(x) \neq f(x)$$

  - Processor failures, biased gradient estimates.

$$\tilde{f}(x^0 + \delta e_i) = \begin{cases} f(x^0 + \delta e_i), & \forall i \neq i^* \\ f(x^0 + \delta e_{i^*}) & \text{w.p. } p \text{ for } i = i^* \\ V & \text{w.p. } 1-p \text{ for } i = i^*, \end{cases}$$

$$g(x^0,\varepsilon)_i = \frac{1}{\delta}[f(x^0 + \delta e_i) - f(x^0)], \quad E(g(x^0,\varepsilon)) \neq \nabla_\delta f(x^0)$$

Our algorithm and convergence analysis

Deterministic trust region framework

1. Compute a potential step

$m_k(x) = f(x^k) + g_k^\top (x - x^k) + \frac{1}{2}(x - x^k)^\top H_k (x - x_k)$ in $B(x_k, \Delta_k)$.

(g_k, H_k are some gradient and Hessian approximations at x^k.)

Compute a point x^+ which minimizes (reduces) $m(x)$ in $B(x^k, \Delta_k)$.

2. Check decrease

Compute $f(x^+)$ and check if f is reduced comparably to m by x^+.

3. Successful step

If yes $x^{k+1} := x^+$, set $\Delta_{k+1} \geq \Delta_k$.

Generate new g_{k+1}, H_{k+1}.

4. Unsuccessful step

Otherwise, $x_{k+1} := x_k$.

Either $\Delta_{k+1} = \gamma^{-1}\Delta_k$, $\gamma > 1$.

Or generate new g_{k+1}, H_{k+1}.

Powell, Conn, S. Toint, Vicente, Wild, Morales.

"Randomized" trust region framework

1. Compute a potential step

$m_k(x) = f(x^k) + g_k^\top (x - x^k) + \frac{1}{2}(x - x^k)^\top H_k (x - x_k)$ in $B(x_k, \Delta_k)$.

(g_k, H_k are some random gradient and Hessian approximations at x^k)

Compute a point x^+ which minimizes (reduces) $m(x)$ in $B(x^k, \Delta_k)$.

2. Check decrease

Compute $f(x^+)$ and check if f is reduced comparably to m by x^+.

3. Successful step

If yes $x^{k+1} := x^+$, set $\Delta_{k+1} = \gamma \Delta_k$.

Generate new g_{k+1}, H_{k+1}.

4. Unsuccessful step

Otherwise, $x_{k+1} := x_k$, set $\Delta_{k+1} = \gamma^{-1} \Delta_k$.

Generate new g_{k+1}, H_{k+1}.

"Refreshing" models at each iteration allows the occasional use of really bad models.

Bandeira, S., Vicente,'12
Cartis, S., '14

"Stochastic" trust region framework

1. Compute a potential step

$m_k(x) = f(x^k) + g_k^\top (x - x^k) + \frac{1}{2}(x - x^k)^\top H_k (x - x_k)$ in $B(x_k, \Delta_k)$.
(g_k, H_k are some random gradient and Hessian approximations at x^k).
Compute a point x^+ which minimizes (reduces) $m(x)$ in $B(x^k, \Delta_k)$.

2. Check decrease

Compute random estimates $f_k \approx f(x^k)$ and $f_k^+ \approx f(x^+)$
and check if $f_k - f_k^+$ is comparable to $m(x^k) - m(x^+)$.

3. Successful step

If yes $x^{k+1} := x^+$, set $\Delta_{k+1} = \gamma \Delta_k$.
Generate new g_{k+1}, H_{k+1}.

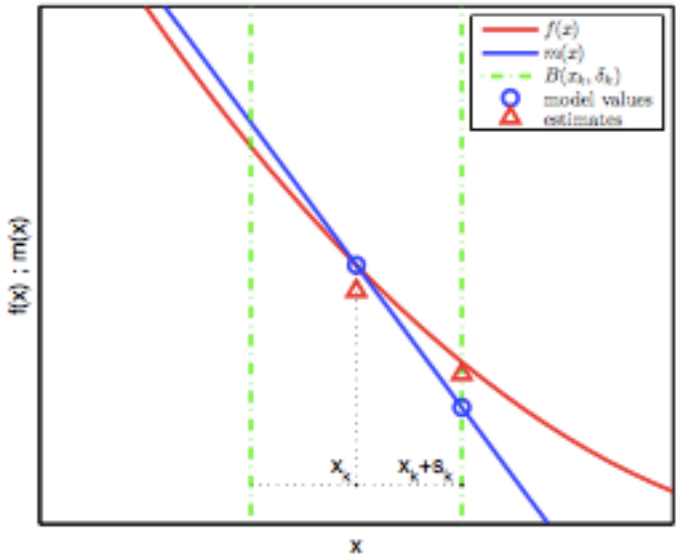
4. Unsuccessful step

Otherwise, $x_{k+1} := x_k$, set $\Delta_{k+1} = \gamma^{-1} \Delta_k$ by the constant factor.
Generate new g_{k+1}, H_{k+1}.

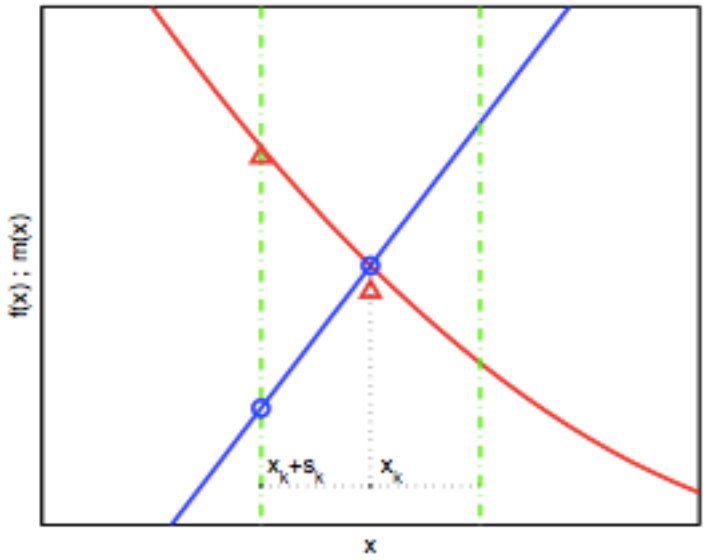
"Refreshing" function estimates at each iteration allows the occasional use of really bad function values.

(Chen, Menickelly, S. 2015)

What can happen on each step

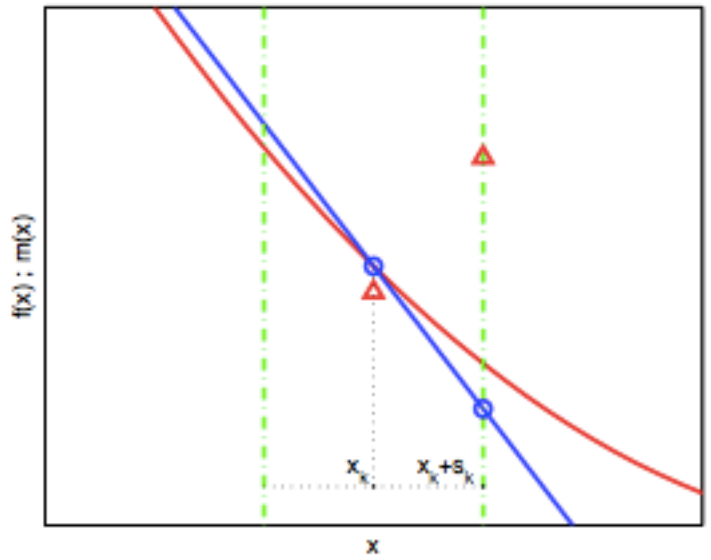


(a) Good model; good estimates.
True successful steps.

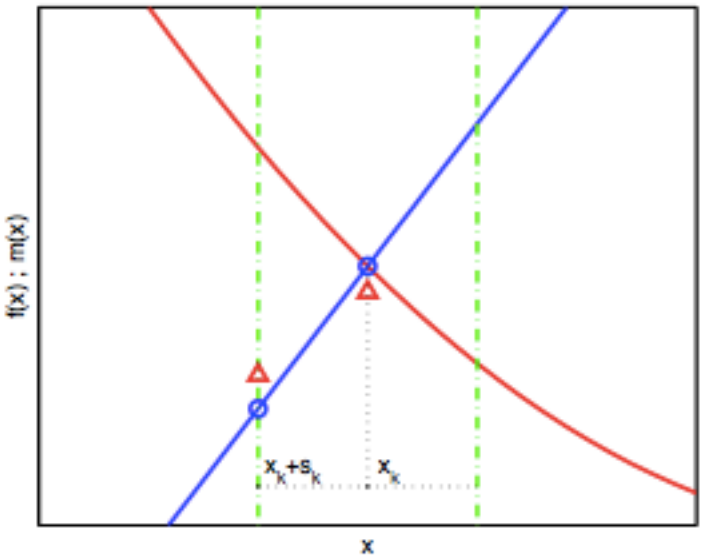


(b) Bad model; good estimates.
Unsuccessful steps.

What can happen at each step



(c) Good model; bad estimates.
Unsuccessful steps.



(d) Bad model; bad estimates.
False successful steps: f can increase!

What do we need from random models and random function estimates?

We need likely Taylor-like behavior of first-order models

For some constant κ, $\|\nabla f(x^k) - g_k\| \leq \kappa \Delta_k$,
with probability at least $1 - \alpha$, conditioned on the past.

We need likely accuracy from the function estimates

For some small enough constant θ,
$\max\{|\tilde{f}(x^k) - f(x^k)|, |\tilde{f}(x^k + s^k) - f(x^k + s^k)|\} \leq \theta \Delta_k^2$,
with probability at least $1 - \beta$, conditioned on the past.

Model and estimate accuracy depends on Δ_k – the TR radius

Probabilities α and β are constant throughout the algorithm

Key ideas in establishing convergence

Construct the following stochastic process

$$\Phi_k = \nu f(X_k) + (1-\nu)\Delta_k^2$$

where $\nu \in (0,1)$ is a fixed constant (carefully selected)

Φ_k is bounded from below

Prove:

$$E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}] \leq -\sigma \Delta_k^2 < 0.$$

Hence:

Φ_k is a supermartingale and $\Delta_k \to 0$

Key steps of analysis

Assume m_k is always κ-fully-linear
and estimates f_k and f_k^+ are always θ-accurate.

⇩

There exists a constant C (dependent on algorithmic parameters and Lipschitz constants):

If $\Delta_k \geq \gamma C$, then either
$\Delta_{k+1} = \Delta_k / \gamma$ or $f(x_{k+1}) \leq f(x_k) - h(\Delta_k)$

If $\Delta_k \leq C$, then always
$\Delta_{k+1} = \gamma \alpha_k$ and $f(x_{k+1}) \leq f(x_k) - h(\Delta_k)$

⇩

Guaranteed constant decrease in $f(x_k)$ once Δ_k decreases below some threshold

Key steps of analysis

Assume m_k is ~~always~~ κ-fully-linear and estimates f_k and f_k^+ are ~~always~~ θ-accurate.

w.p. $\geq 1-\alpha\beta$

There exists a constant C (dependent on algorithmic parameters and Lipschitz constants):

If $\Delta_k \geq \gamma C$, then ~~either~~ $\Delta_{k+1} = \Delta_k/\gamma$ or ~~$f(x_{k+1}) \geq f(x_k) + h(\Delta_k)$~~

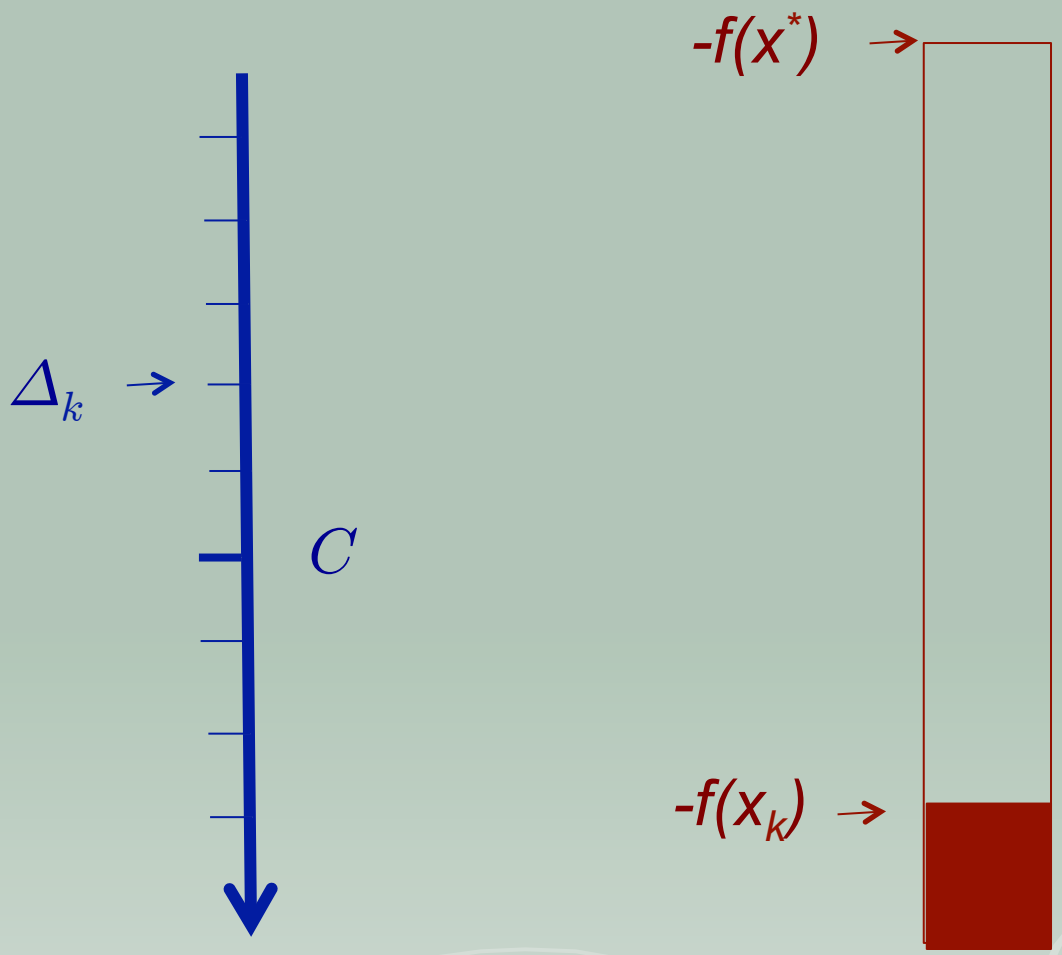
w.p. $\geq 1-\alpha\beta$

If $\Delta_k \leq C$, then ~~always~~ $\Delta_{k+1} = \gamma\alpha_k$ and $f(x_{k+1}) \geq f(x_k) + h(\Delta_k)$

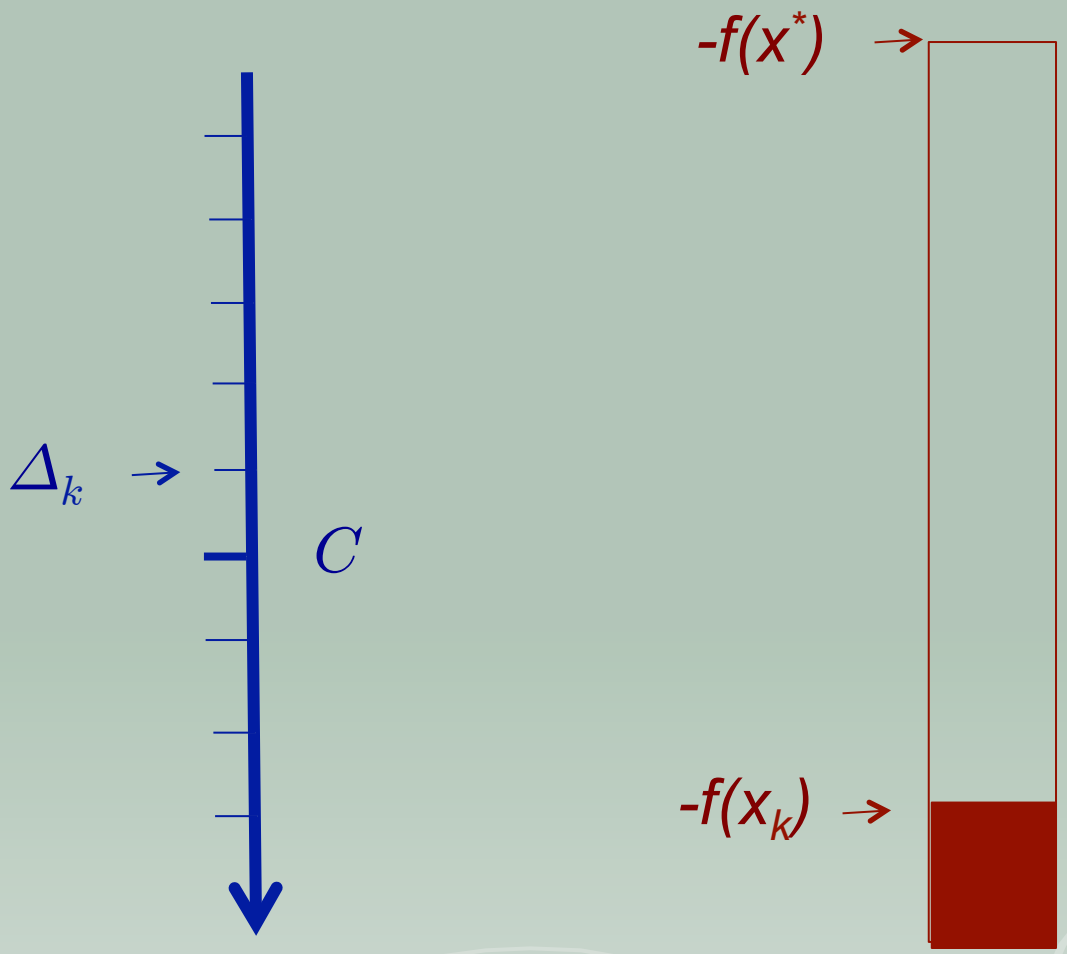
w.p. $\geq 1-\alpha\beta$

Different behavior depending on Δ_k being larger γC *or* smaller that *C*. $f(x_k)$ increases w. prob. $\geq 1-\alpha\beta$ when $\Delta_k \leq C$

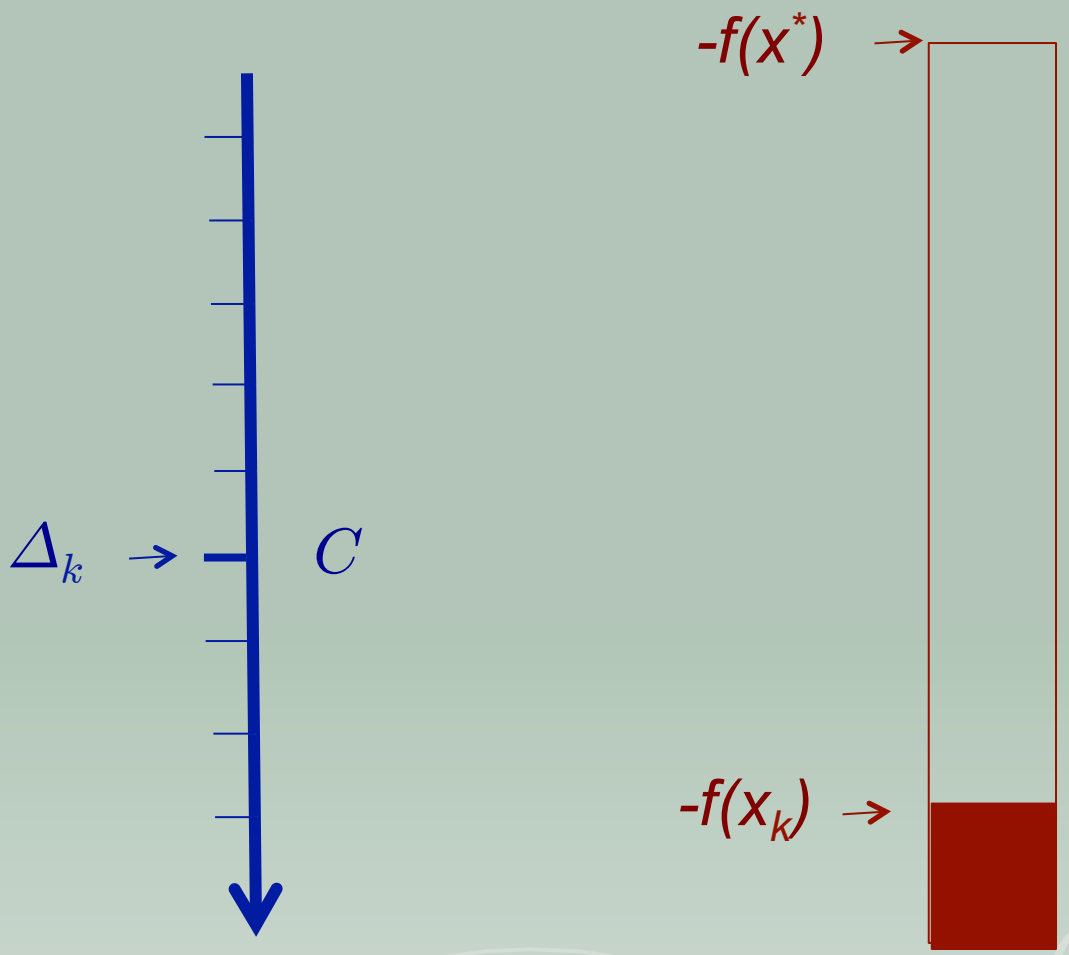
Illustration



Illustration

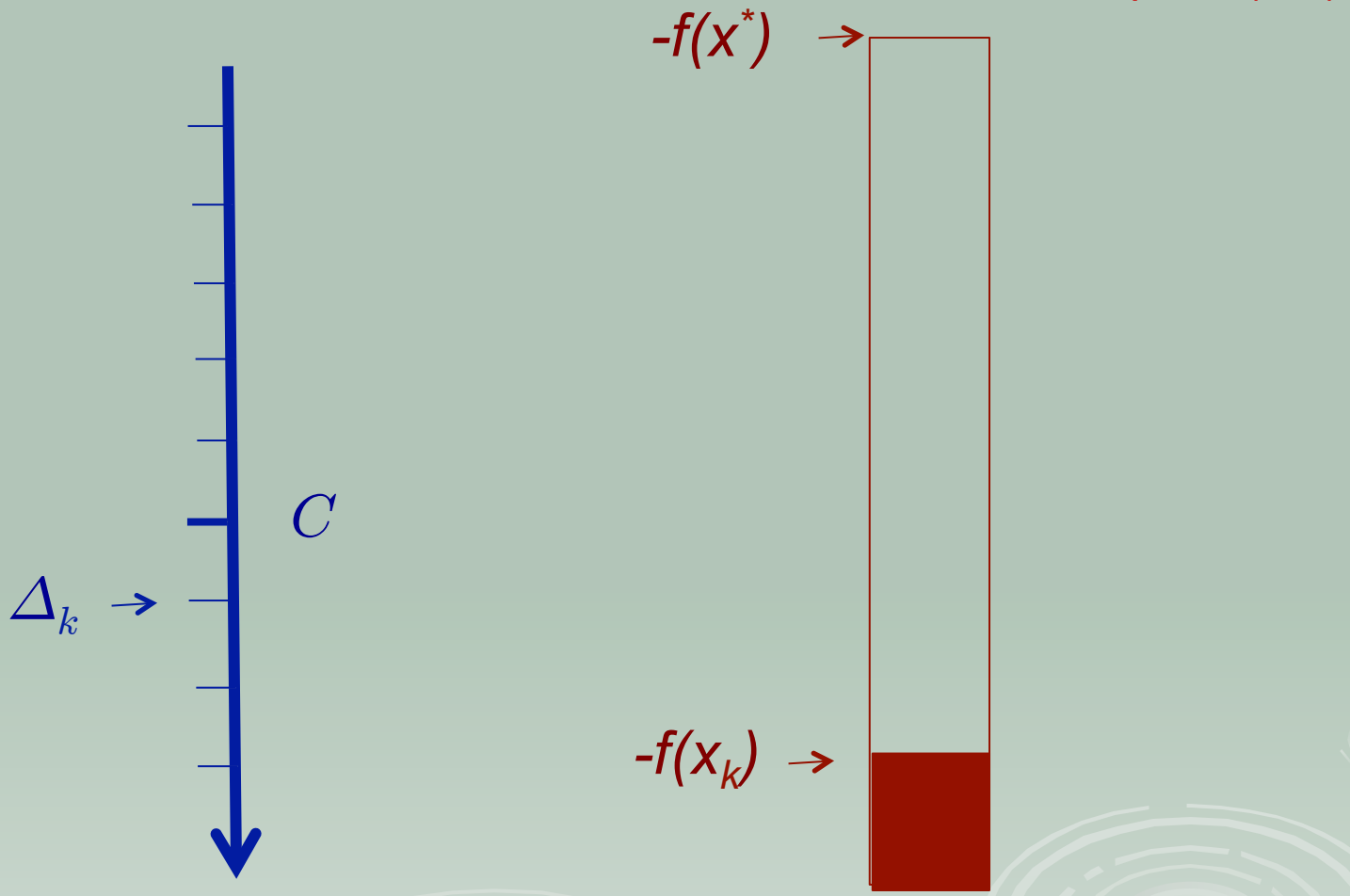


Illustration



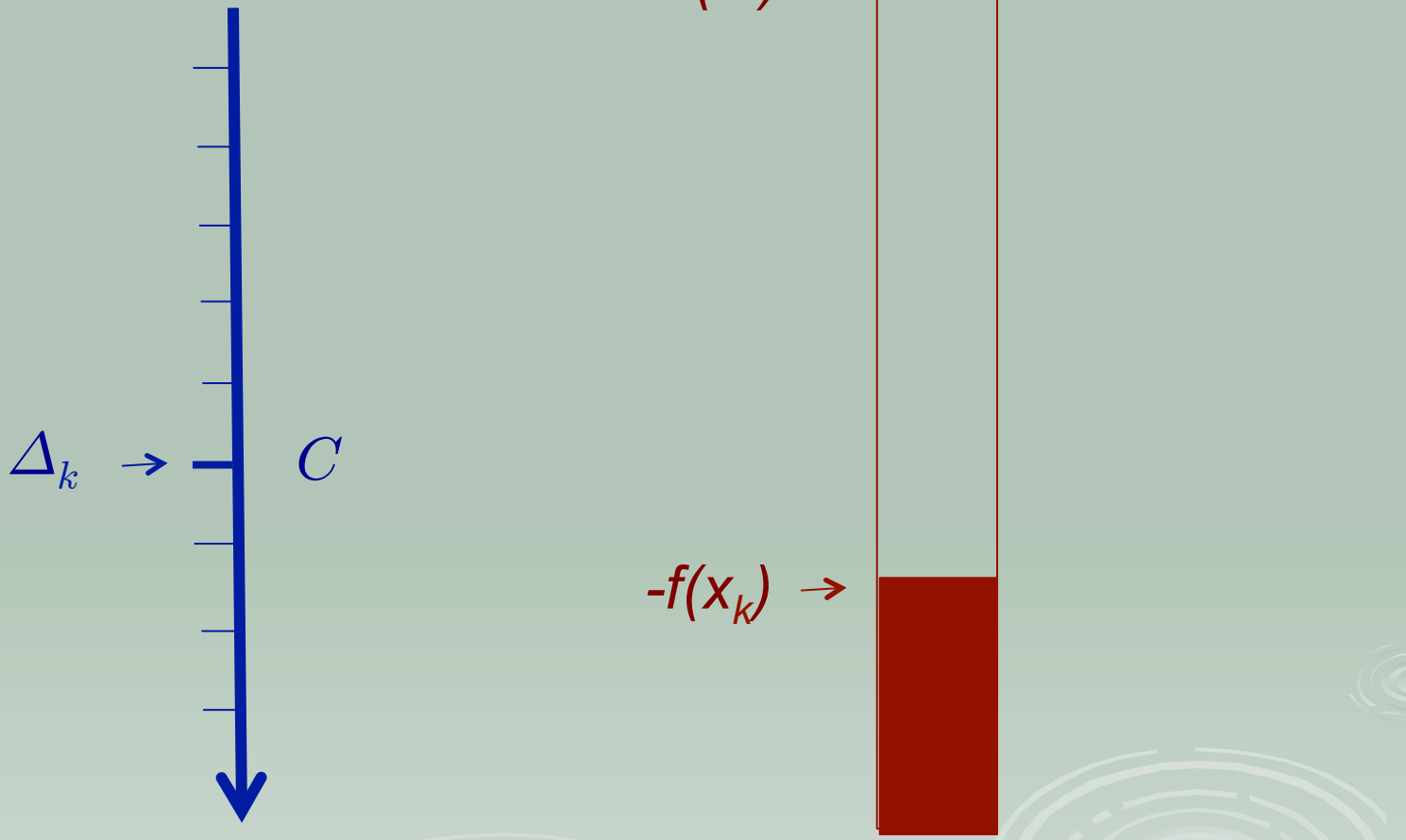
Illustration

w.p.$\leq \alpha(1\text{-}\beta)$



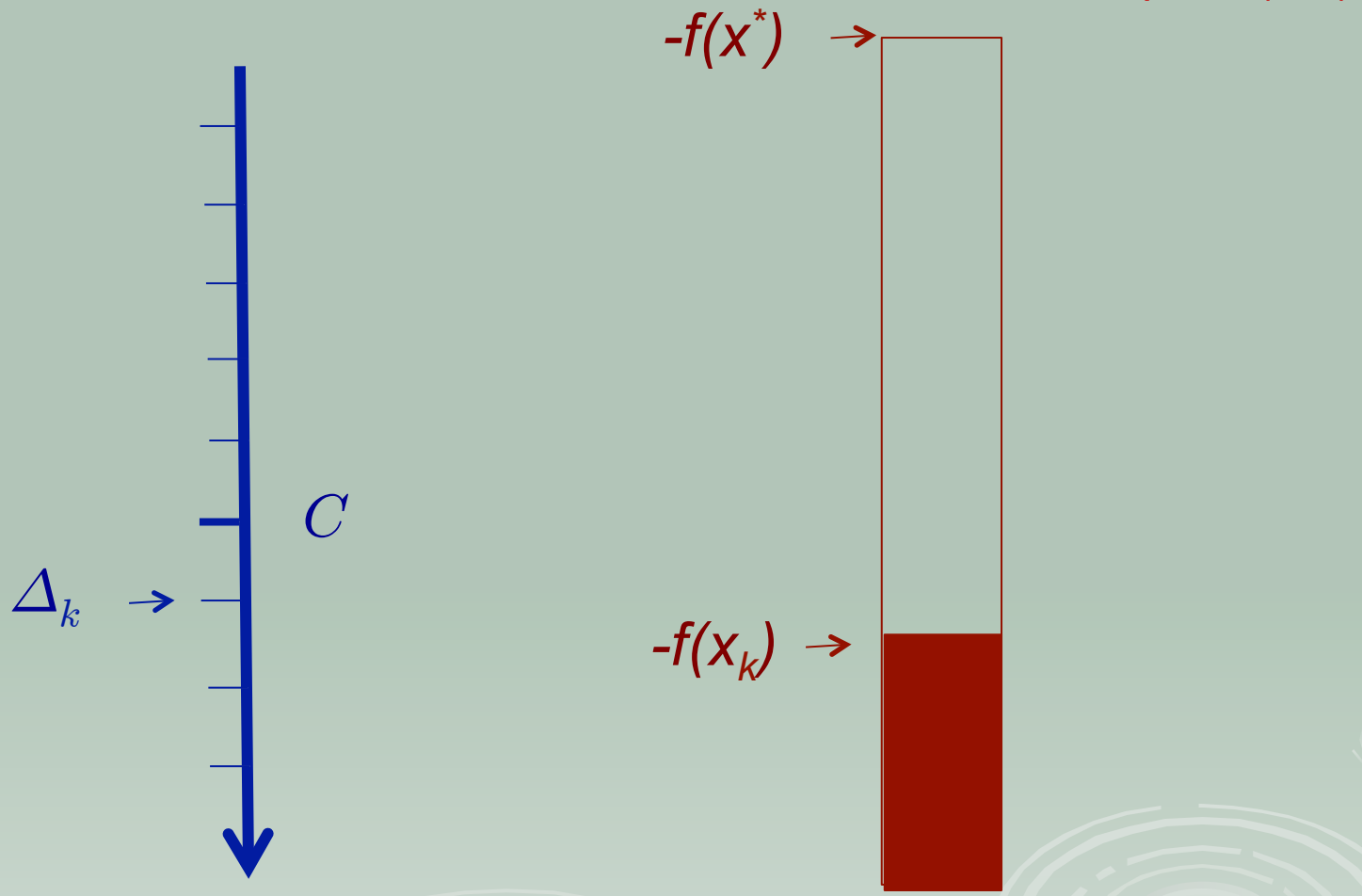
Illustration

w.p.$\geq$(1- α) (1-β)



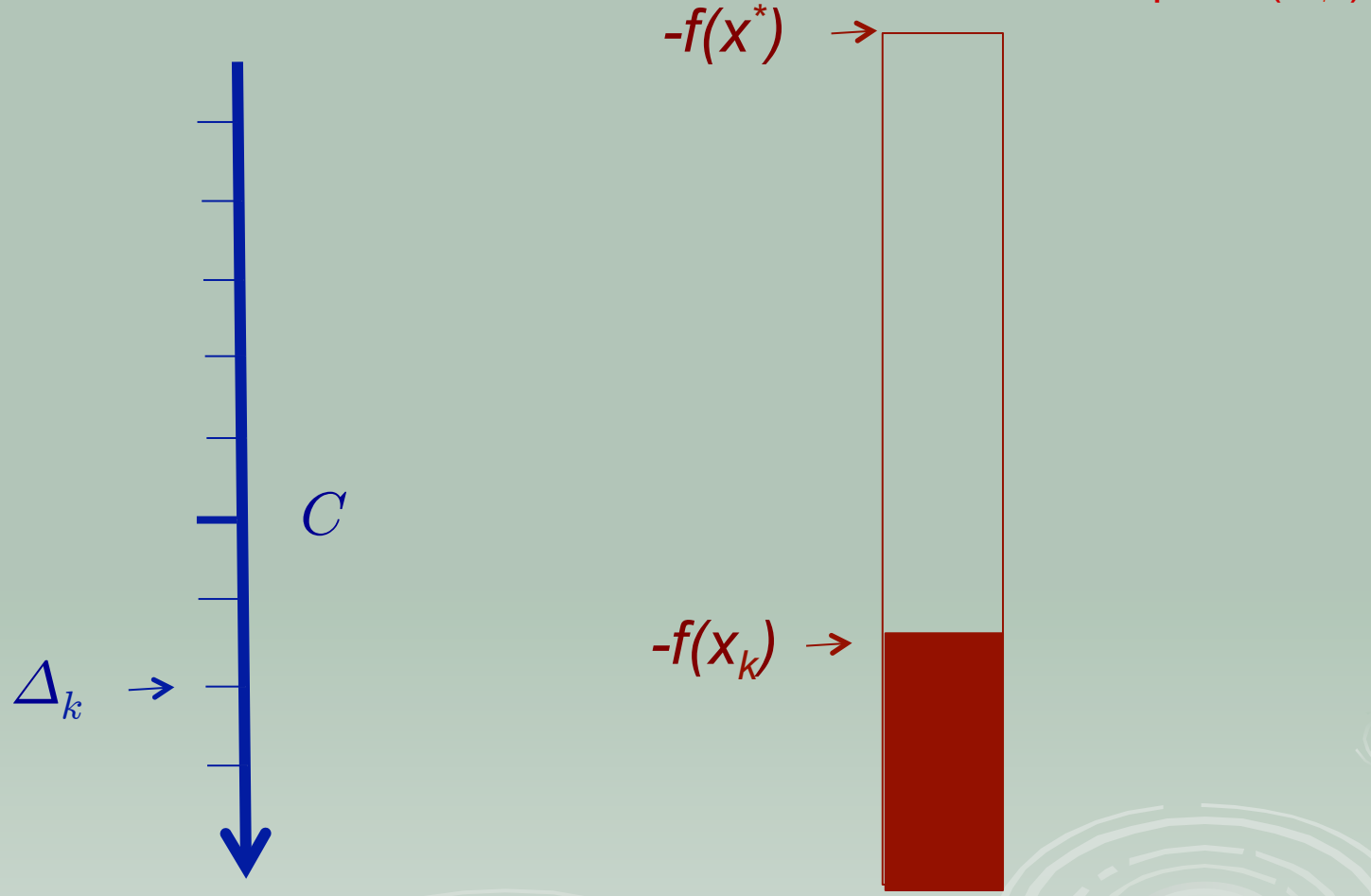
Illustration

w.p.$\leq \alpha(1\text{-}\beta)$



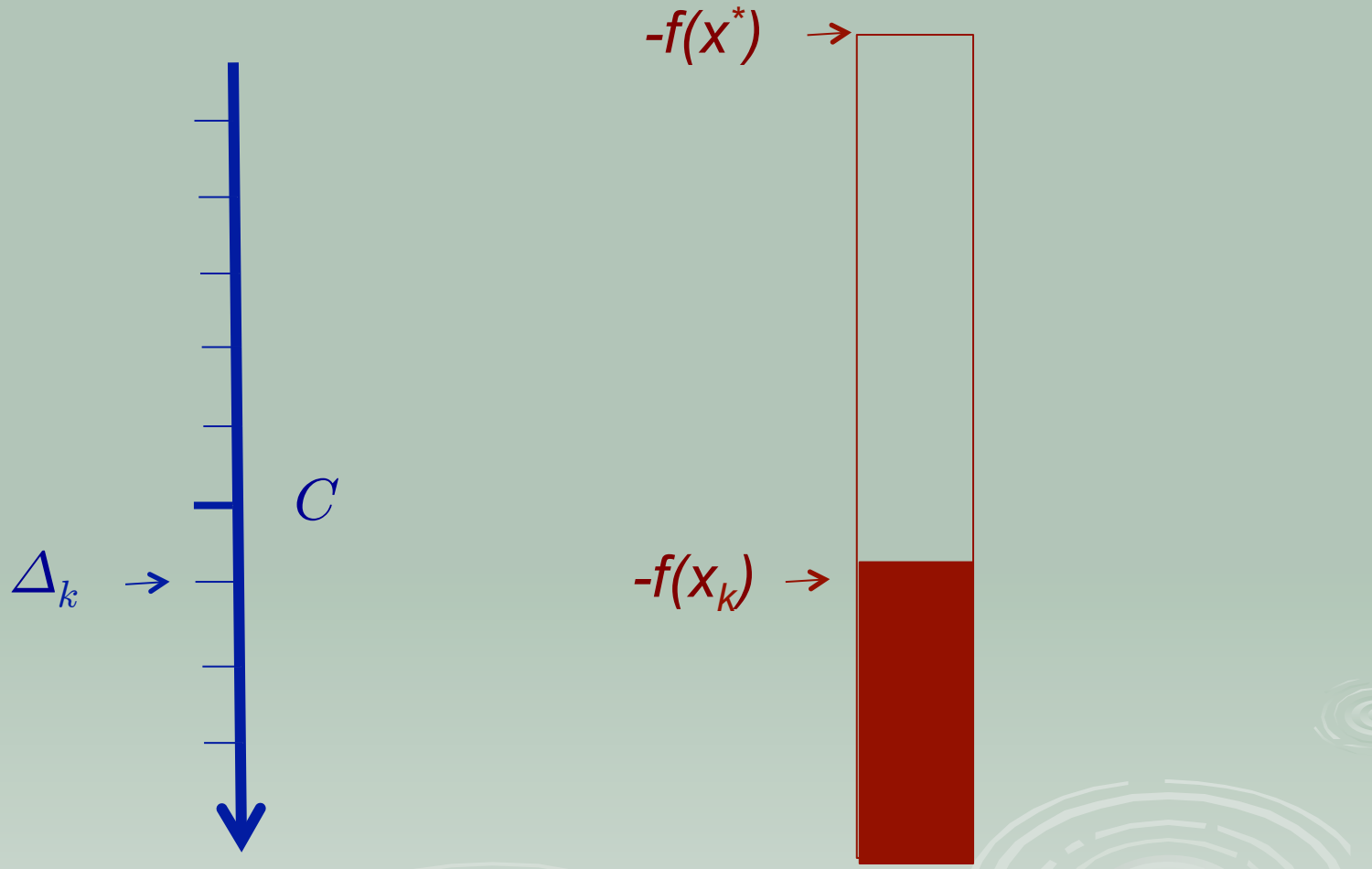
Illustration

w.p.$\leq \alpha(1-\beta)$

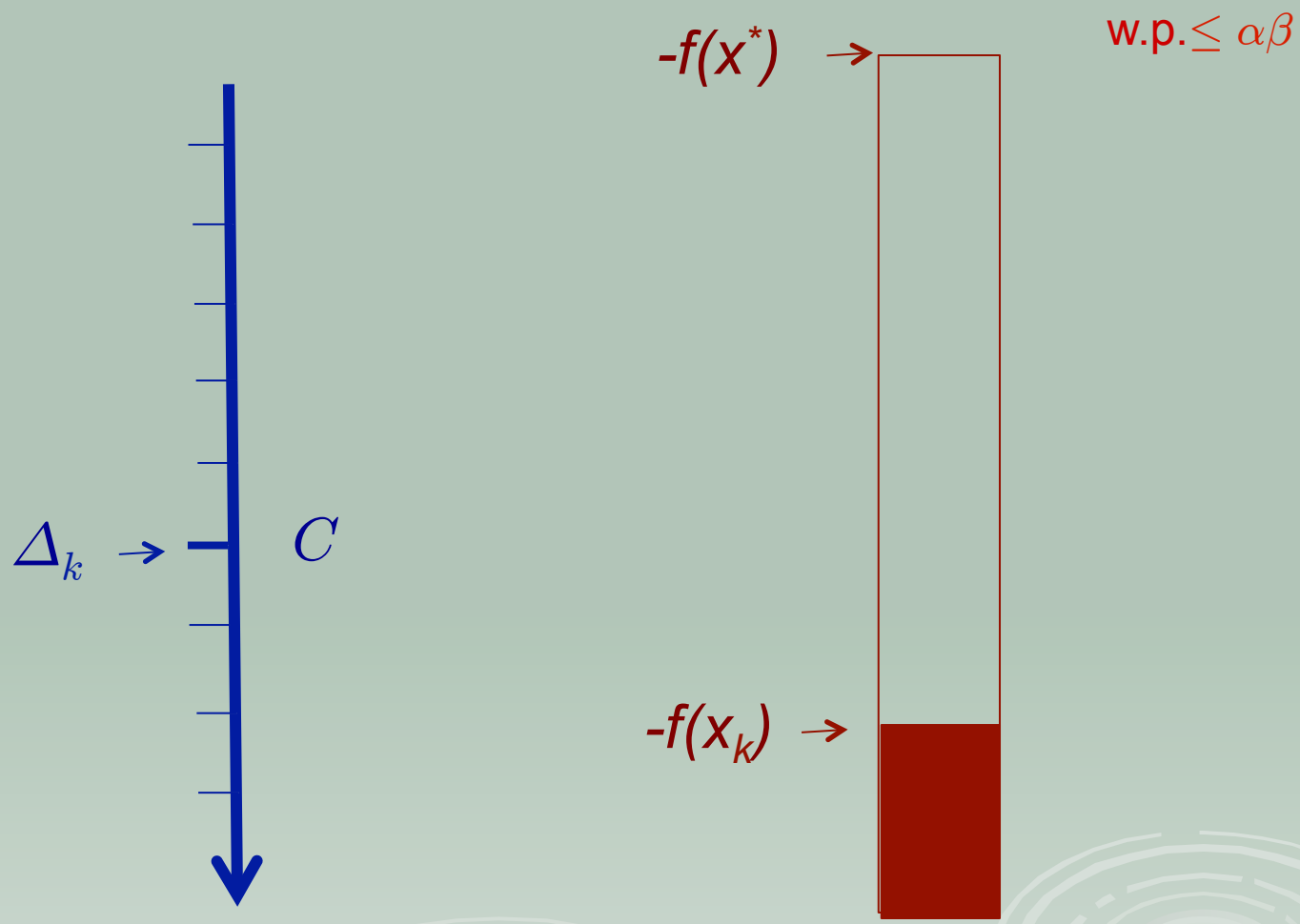


Illustration

w.p.$\geq$(1- α) (1-β)



Illustration



Main convergence result

Theorem: There exists a constant $p \geq 0$, dependent on f and algorithmic constants, such that if

$$\alpha\beta \leq p \text{ and } (1-\alpha)(1-\beta) \geq \frac{1}{2}$$

$\|\nabla f(x^k)\| \to 0$ with probability 1. (Chen, Menickelly, S. 2014)

Specifically,

$$\frac{\alpha\beta}{(1-\alpha)(1-\beta)} \leq \mathcal{O}(1), \quad \alpha\beta \leq \mathcal{O}(\frac{1}{L}), \quad \theta \leq \mathcal{O}(L),$$

where L is the Lipschitz constant of f

Computational results

Biased and unbiased noise examples, again.

- Noisy function samples.
  - Unbiased nose $E(f(x,\varepsilon)) = f(x) \quad \forall x$

$$f(x,\varepsilon) = f(x) + \varepsilon, \text{or} f(x,\varepsilon) = f(x)(1+\varepsilon),\ E(\varepsilon) = 0$$

  - Biased noise

$$\tilde{f}(x) = f(x,\varepsilon) = \begin{cases} f(x), & \text{w.p. } p \\ \varepsilon(x) \leq V & \text{w.p. } 1-p, \end{cases}$$

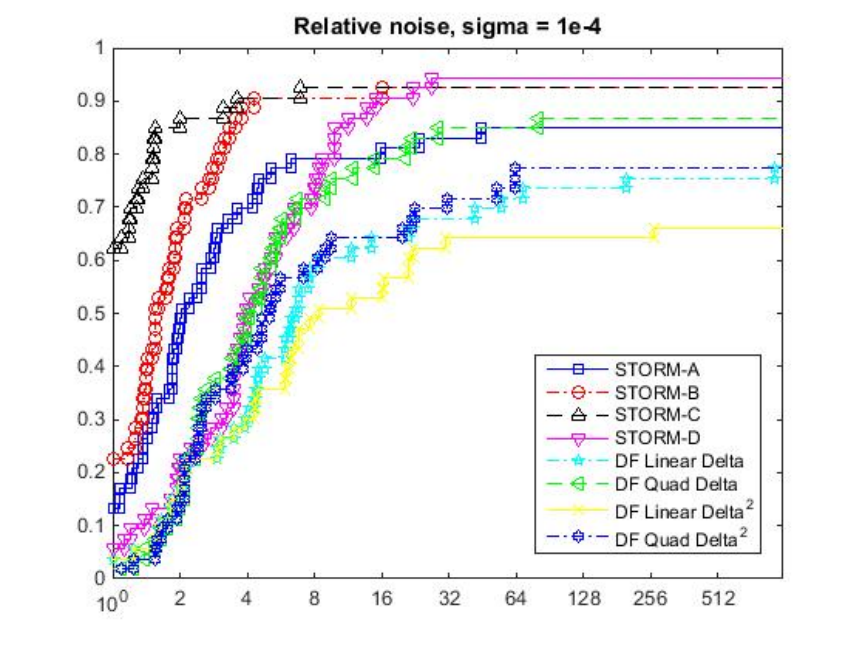
$$E(f(x,\epsilon)) = pf(x) + (1-p)\varepsilon(x) \neq f(x)$$

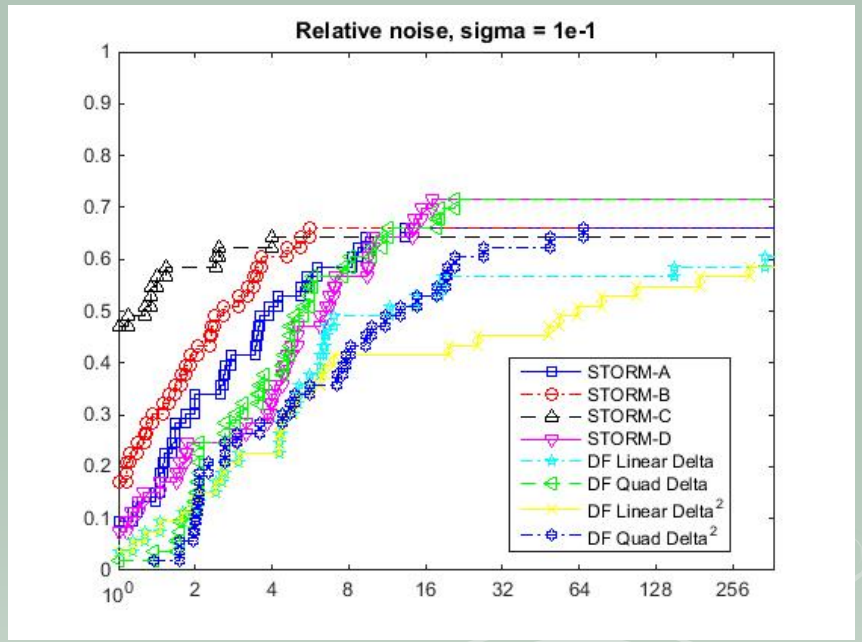
  - Processor failures, biased gradient estimates.

$$\tilde{f}(x^0 + \delta e_i) = \begin{cases} f(x^0 + \delta e_i), & \forall i \neq i^* \\ f(x^0 + \delta e_{i^*}) & \text{w.p. } p \text{ for } i = i^* \\ V & \text{w.p. } 1-p \text{ for } i = i^*, \end{cases}$$

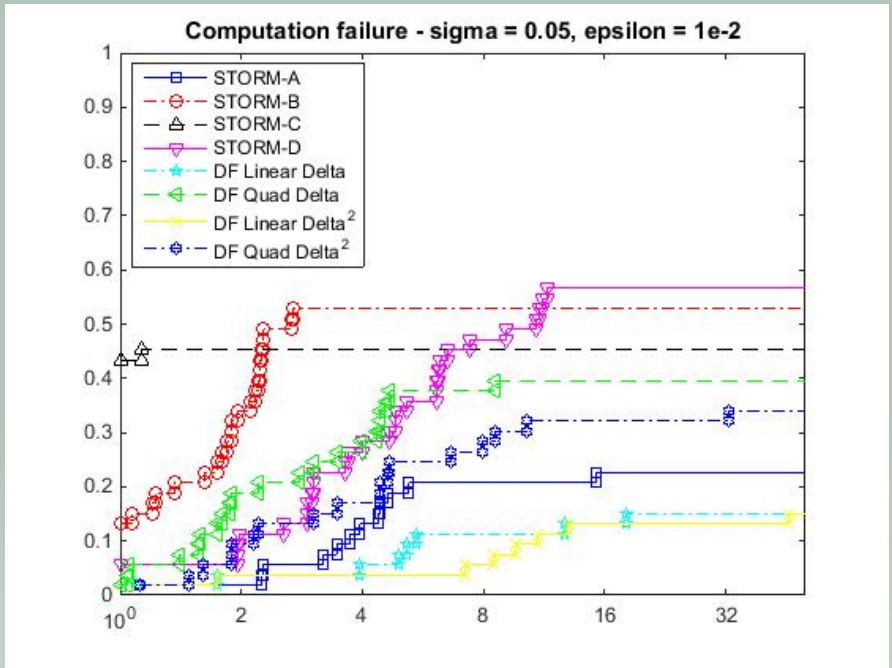
$$g(x^0,\varepsilon)_i = \frac{1}{\delta}[f(x^0 + \delta e_i) - f(x^0)], \quad E(g(x^0,\varepsilon)) \neq \nabla_\delta f(x^0)$$

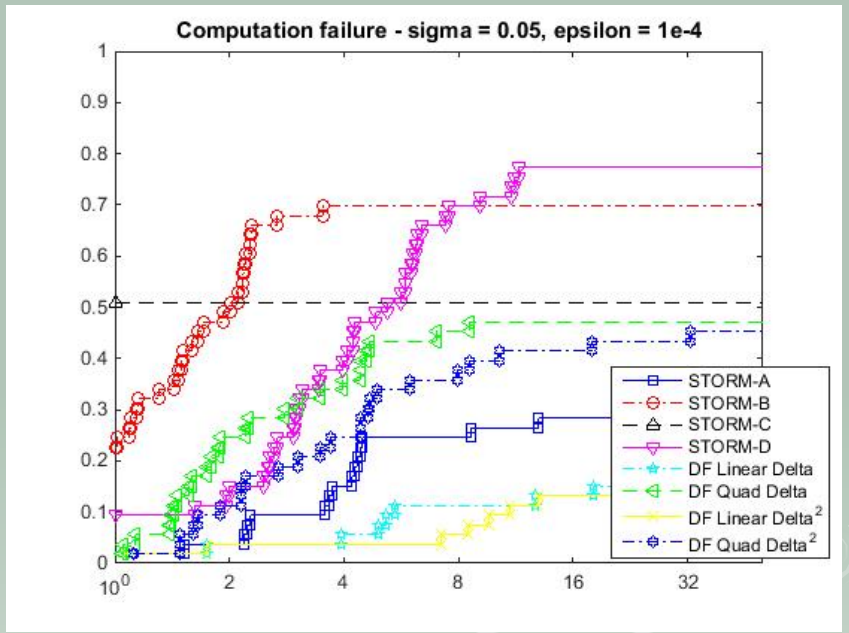
Comparison of STORM with sample averaging TR method. The relative noise case



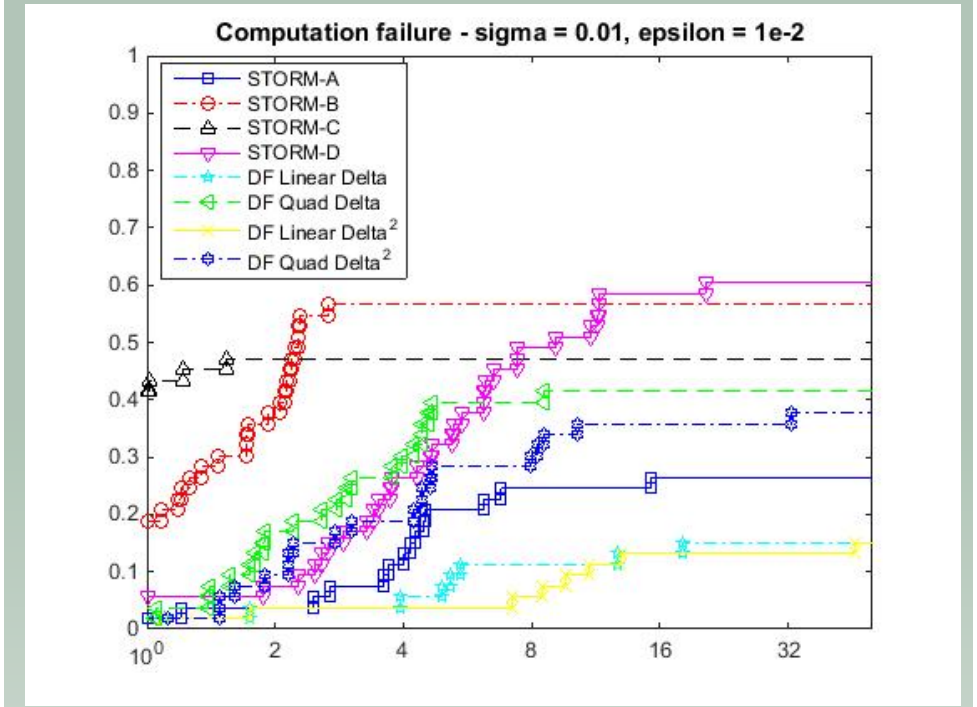


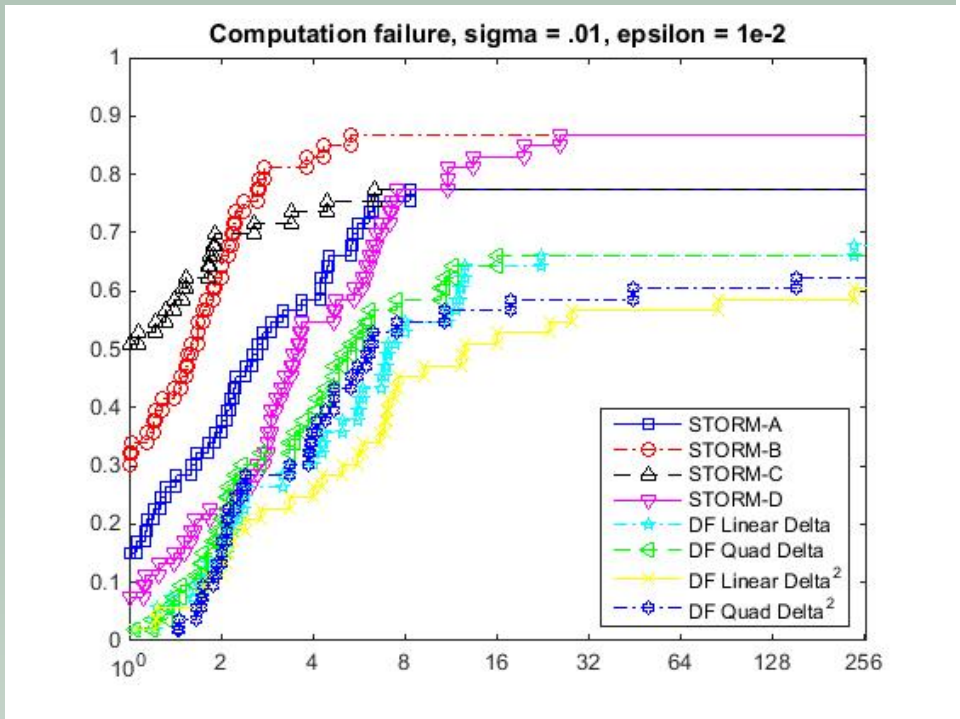
Comparison of STORM with sample averaging TR method. The computations failures



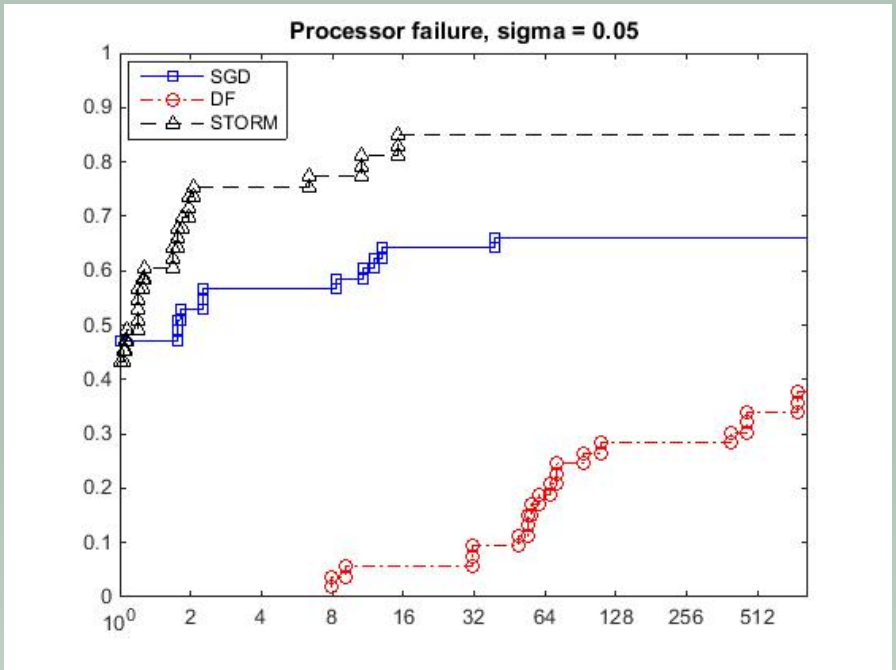


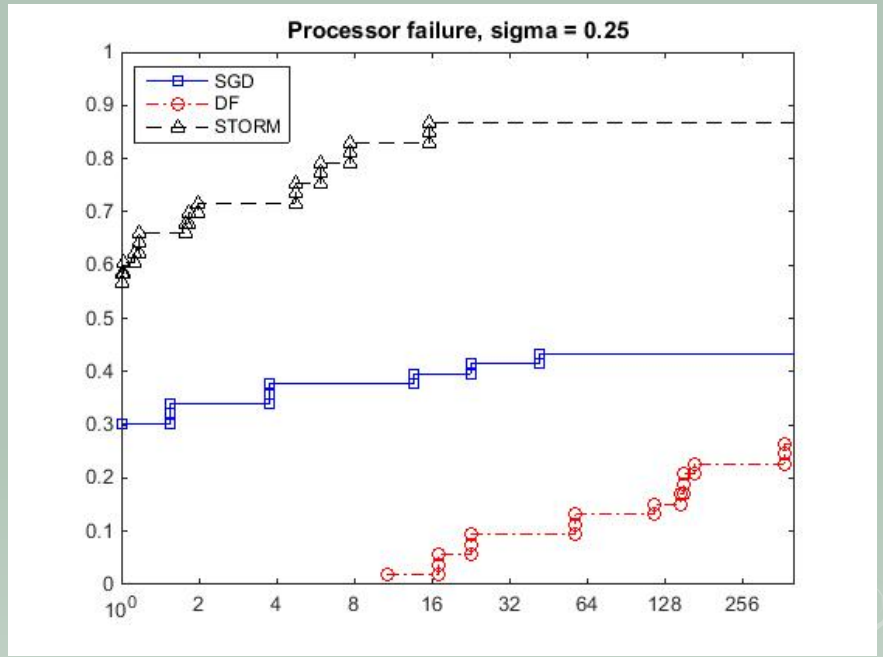
Comparison of STORM with sample averaging TR method. The computations failures



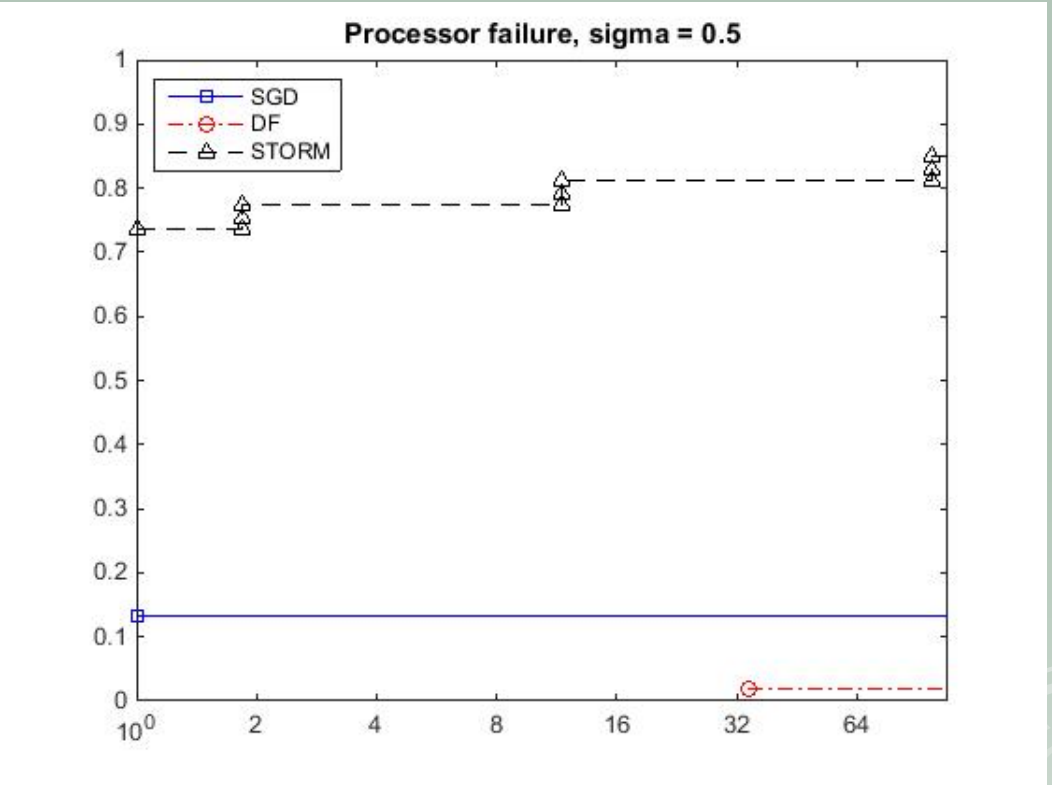


Comparison of STORM with sample averaging TR method. The processor failures





Comparison of STORM with sample averaging TR method. The computations failures



Conclusion

- We propose a general framework for stochastic inexact first (and second) order methods.
  - No assumption on distribution, or expectation.
  - Models are suff. accurate with constant probability.
  - Applies to standard deterministic frameworks.
  - Applies to cases of biased noise.
  - Works well in practice.
  - Can view this a demonstration of robustness of a standard framework.

Future work

- Convergence rates analysis.
- Sampling rate analysis for randomly sampled models.
- Use of learning guarantees and Rademacher complexity of model classes.
- Extend to convex optimization.
- More examples of models that fit the framework.

Thank you!