Reliability assessment framework for large scale causal logic networks

Nicoleta Cucu Laurenciu & Sorin Cotofana

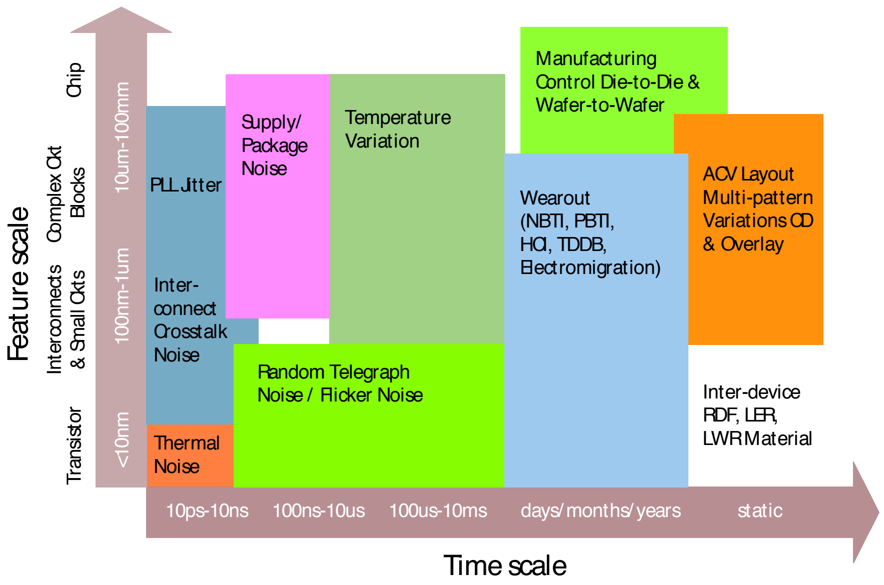
Computer Engineering Laboratory, Delft University of Technology, The Netherlands

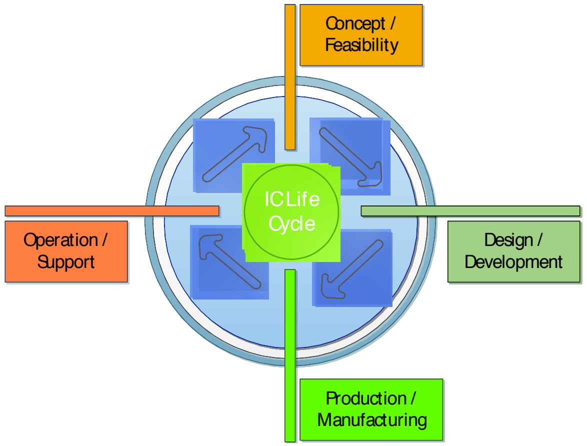
i-RISC Workshop, Bucuresti, September 2013

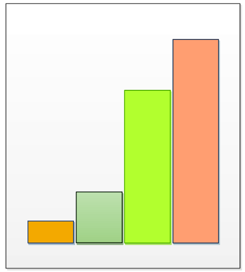
Outline

1. Introduction. General framework.
2. Gate-level reliability characterization.
3. Circuit-level reliability characterization - reliability inference algorithm.
4. Summary and further development.

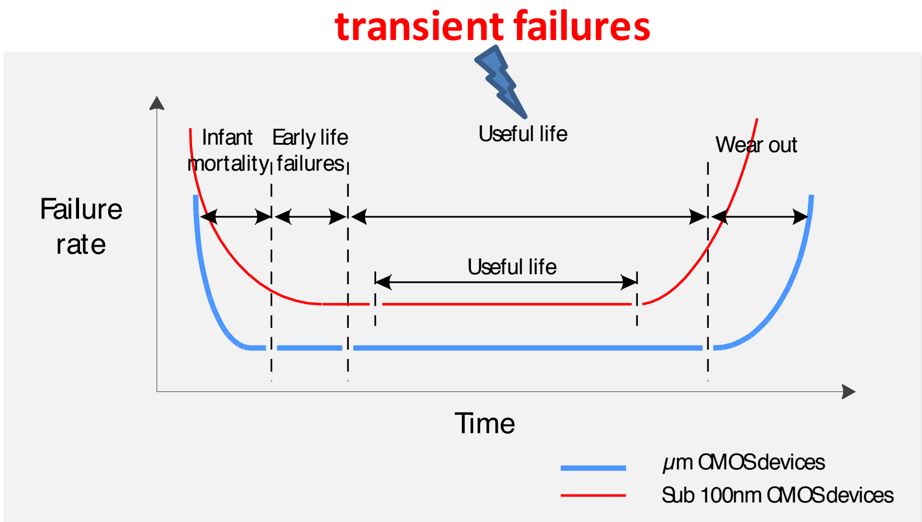
1. IC design-for-reliability impetus







The cost to fix defects / lifecycle stage

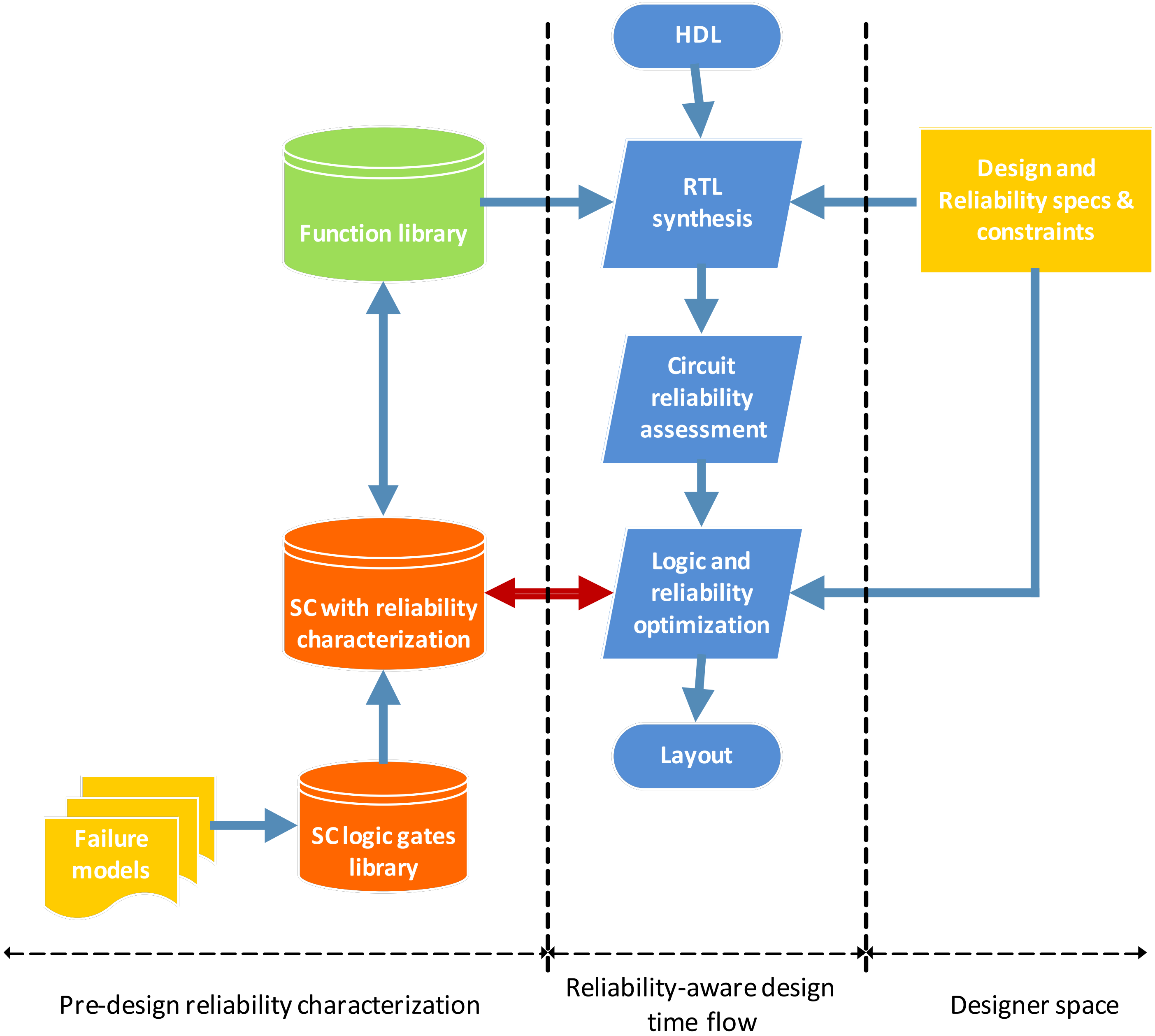


1. IC design-for-reliability – desiderata

1. Evaluate and increase the logical masking capability of ICs via various resiliency techniques (e.g., fault tolerant codecs).
2. Compare architectures and enable a reliability-driven Boolean function synthesis process.

- Fast, yet accurate reliability estimation mechanism which is scalable for tera-scale integrated circuits.
  - Accurate reliability estimation at the gate level – drastic impact on the circuit reliability estimation.
  - Fast approximate reliability estimation at the circuit-level at design-time.

1. Design-for-reliability framework

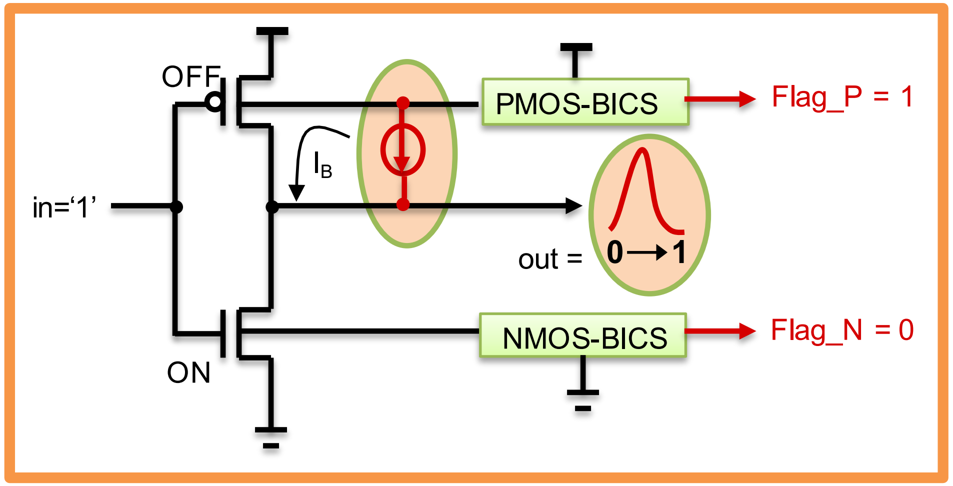


2. SC logic gates reliability pre-characterization

SC with reliability characterization

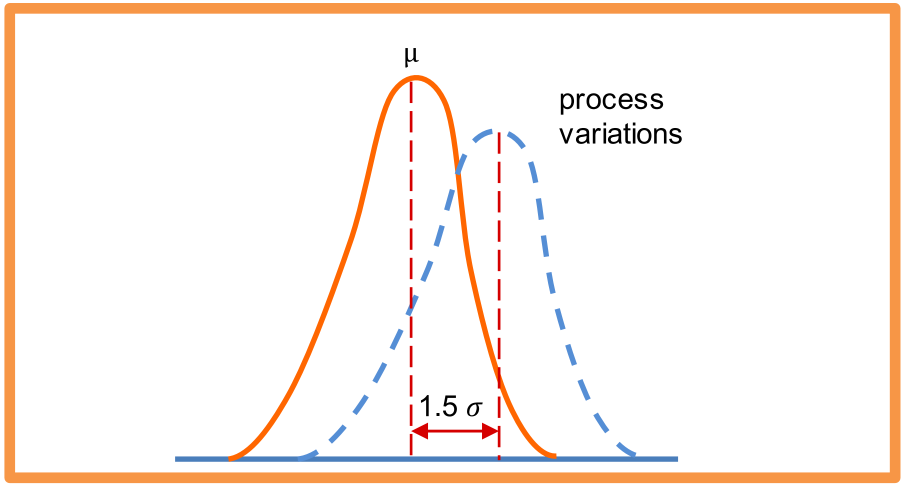
Monte Carlo analysis

Fault macro-modeling



Translate physical defects into equivalent electrical linear (resistors, capacitors) and nonlinear devices (scaled transistors).

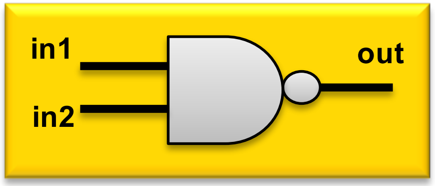
Parameters variations



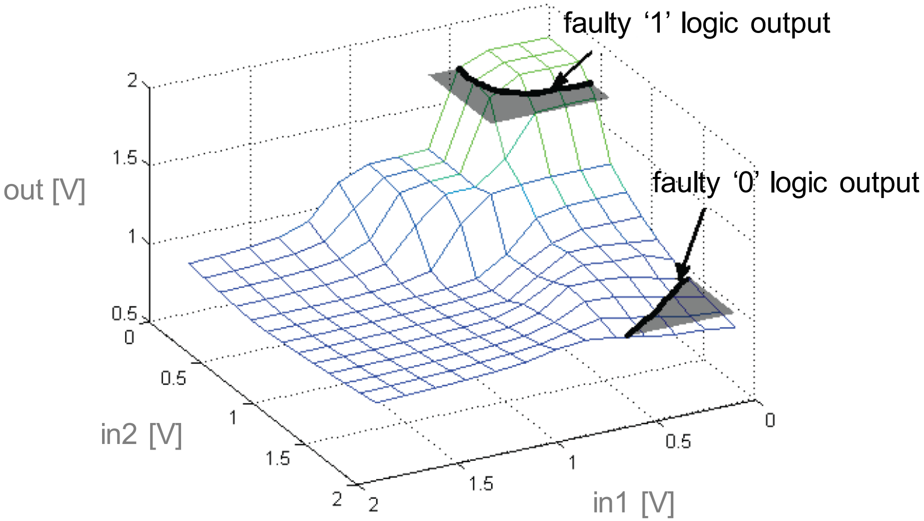
e.g., V_{th} , t_{ox}, L, W

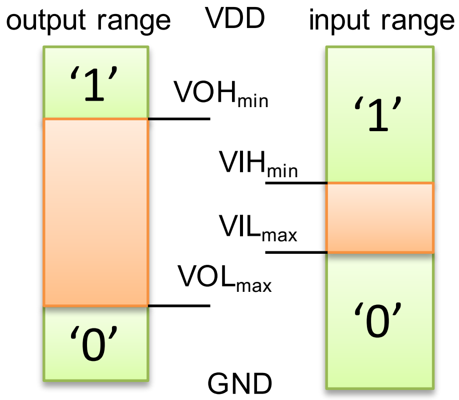
2. SC logic gates reliability pre-characterization

e.g.:



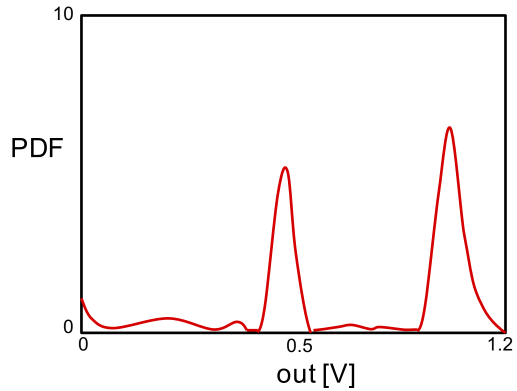
Transfer function surface





Acceptance condition
for correct operation

Probability density function of erroneous output



Monte Carlo analysis

Fault macro-modeling

Parameters variations

3. IC reliability assessment

Problem statement

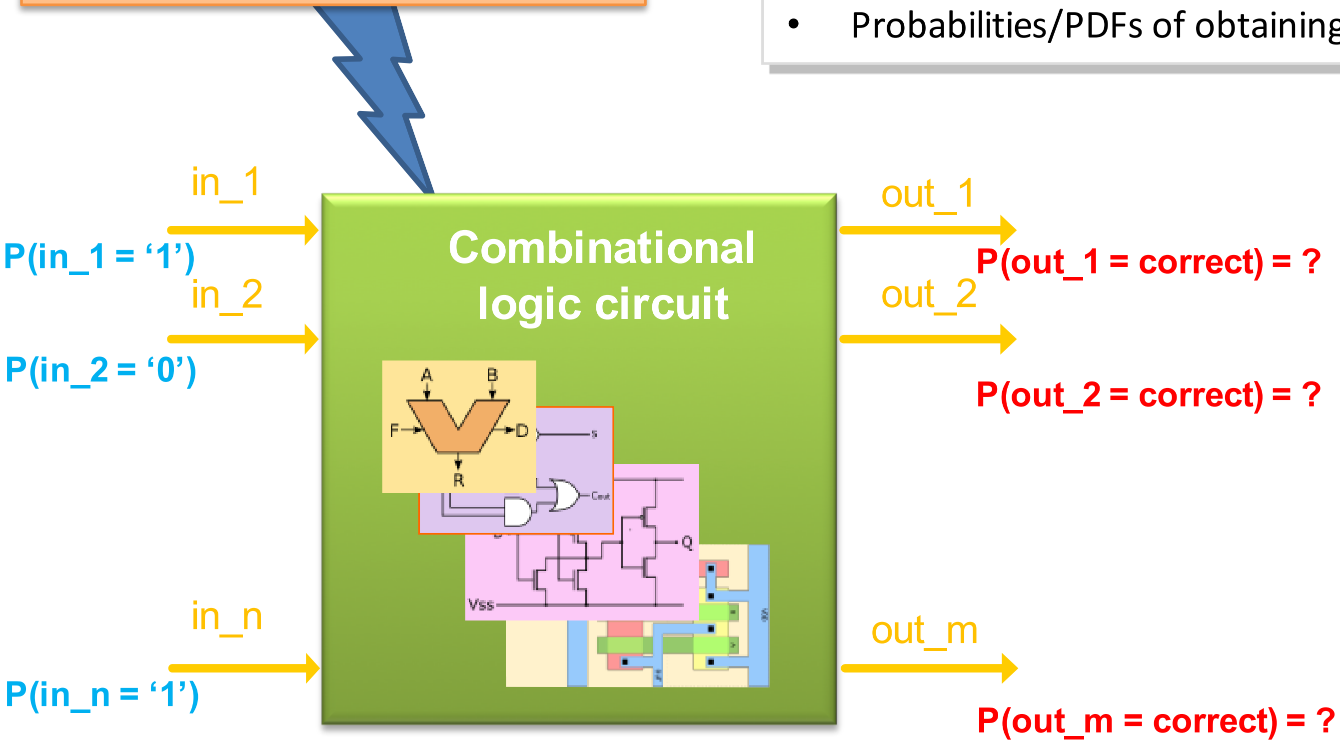
Hypothesis:

- Circuit with given topology and possibly layout;
- Workload, e.g., input vectors and their associated probabilities/PDFs;
- Input aggression profile (environmental – e.g., T, VDD – and fault scenarios - e.g., fault types and their expected probabilities).

Conclusion:

- Probabilities/PDFs of obtaining the correct circuit outputs = ?

Aggression profile:
- environmental (e.g., T, VDD)
- fault scenarios (e.g., $PF_{GATE\ i-j}$, fault types and their expected probabilities)



3. Model formalism (1)

Probabilistic graphical model – Bayesian network

- Elegant framework , which combines:
  - Graph theory – cope with circuit correlations complexity
  - Probability theory – deal with uncertainty

$$p(s_1, \ldots, s_m) = \prod_{i=1}^{m} p(s_i \mid Parents(s_i))$$

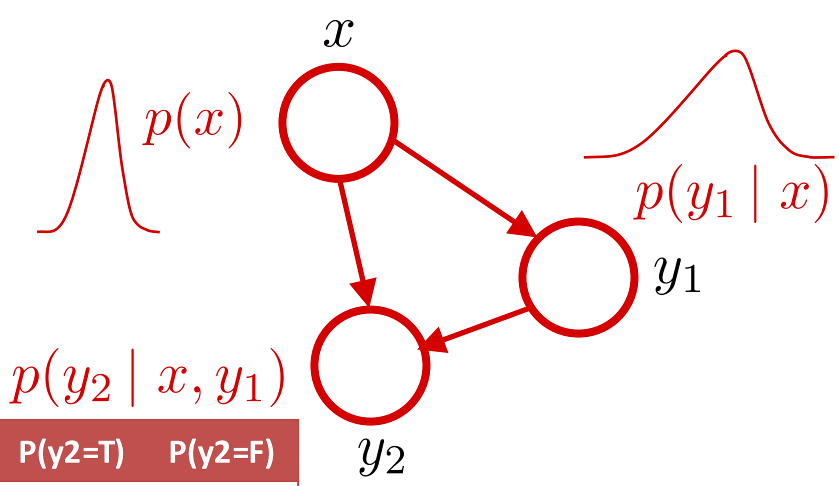
Syntax and semantics:

Directed Acyclic Graph (DAG)

Nodes:

- random variables (logic gates, wires)
- discrete or continuous
- observable or hidden

Edges: direct dependence between nodes



$$p(x, y_1, y_2) = p(y_2 \mid x, y_1)\, p(x, y_1) =$$
$$= p(y_2 \mid x, y_1)\, p(y_2 \mid x)\, p(x)$$

x	y1	P(y2=T)	P(y2=F)
F	F	1.0	0.0
T	F	0.1	0.9
F	T	0.1	0.9
T	T	0.01	0.99

3. Model formalism (2)

Probabilistic graphical model – Bayesian network

- Elegant framework , which combines:
  - Graph theory – cope with circuit correlations complexity
  - Probability theory – deal with uncertainty

$$p(s_1, \ldots, s_m) = \prod_{i=1}^{m} p(s_i \mid Parents(s_i))$$

Syntax and semantics:

Directed Acyclic Graph (DAG)

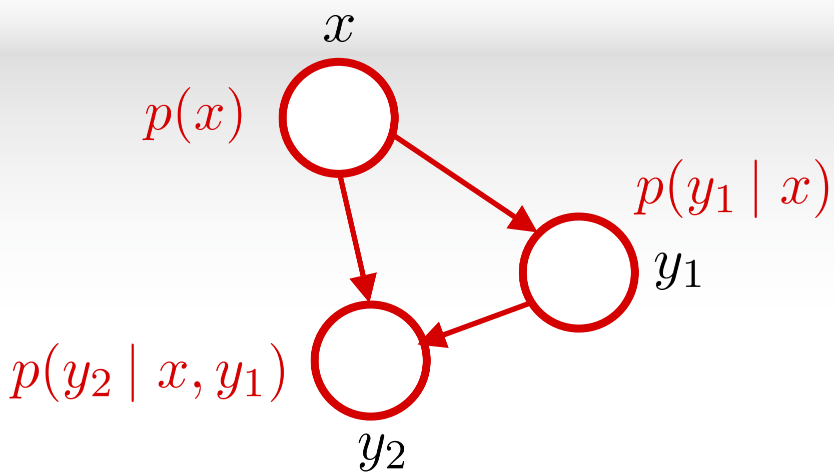
Nodes:

- random variables (logic gates, wires)
- discrete or continuous
- observable or hidden

Edges: direct dependence between nodes

Observable nodes: **x**, known
e.g., the circuit primary inputs
Hidden nodes: **y={y1, y2}**, unknown
endowed with a prior, e.g., the rest of the circuit nodes

$$p(y \mid x) = ? \quad \left(= \frac{p(x, y)}{p(x)} \right)$$



3. Inference engine (1)

Bayes Theorem

$$p(y\,|x) = \frac{p(x \mid y) \cdot p(y)}{p(x)}$$

Posterior probability ?

Likelihood function

Prior probability

Evidence $p(x) = \int p(x \mid y) p(y) dy$

Observable nodes: **x**, known
e.g., the circuit primary inputs
Hidden nodes: **y={y1, y2}**, unknown
endowed with a prior, e.g., the rest of the circuit nodes

Evaluating the posterior $p(y \mid x)$

In practice $p(x)$ is usually intractable to compute, as:

- Closed-form (analytical) solutions are not available;
- Numerical integration is too expensive.

=> necessary to appeal to approximate inference of the posterior.

3. Inference engine (2)

There are two most prominent strategies to approximate inference. They are not mutually exclusive, as they exploit complementary features of the graphical model formalism.

Stochastic inference
in particular sampling

1. Draw samples $y^{(1)} \dots y^{(m)}$ from $p(y \mid x)$
2. Inspect sample statistics

- ✓ Simplicity of implementation
- ✓ Theoretical guarantees of convergence
- ✗ Computationally expensive
- ✗ Convergence hard to diagnose

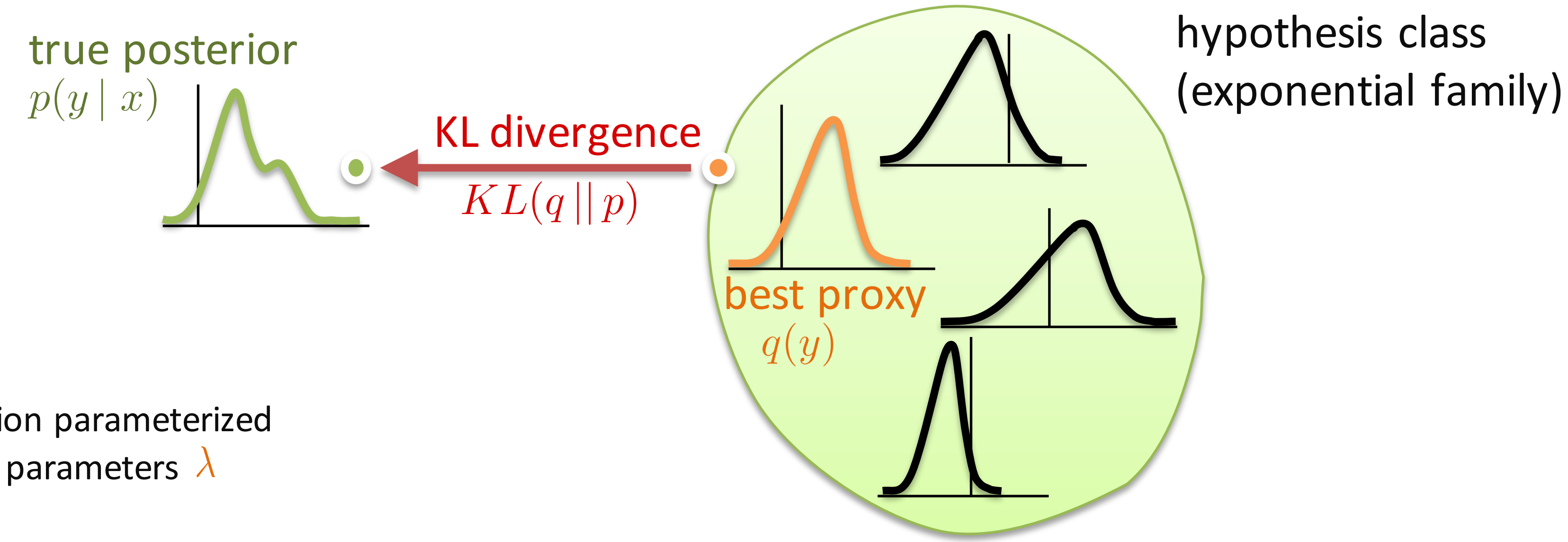
Deterministic variational inference

1. Find an analytical proxy distribution $q(y)$ maximally similar to $p(y \mid x)$
2. Inspect distribution statistics

- ✓ Can lead to simple and accurate approximation procedures
- ✓ Insightful and faster
- ✗ Harder to derive
- ✗ Requires sampling for validation

✗ Neither approach scales easily to the kind of settings encountered in circuit reliability inference.

3. Inference engine (3)



$q(y)$ distribution parameterized by variational parameters λ

Variational approximation

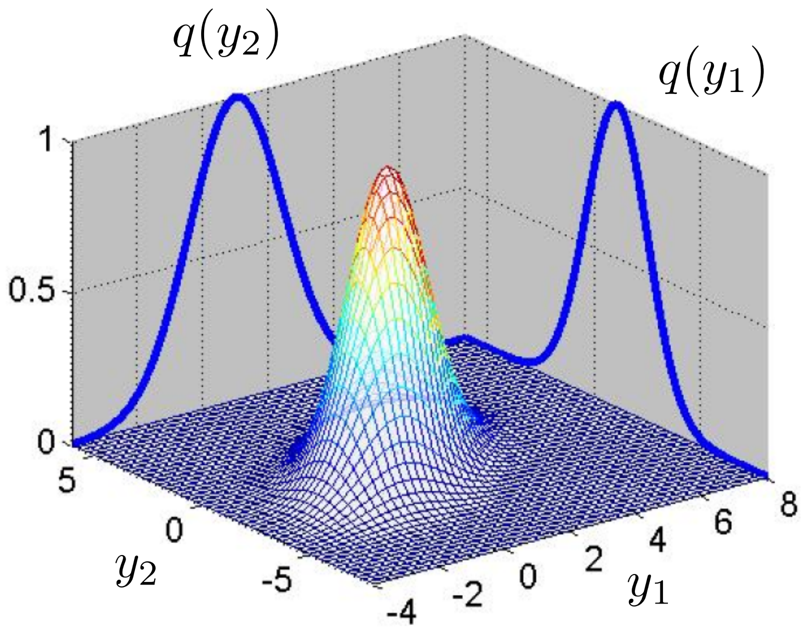
Main idea - cast the posterior inference problem to an optimization problem:

- approximate the posterior $p(y \mid x)$ with a *simpler* distribution $q(y)$ that is *as close as possible*
  - **Simple** = tractable and efficient inference (e.g., factorized distributions typically)
  - **As close as possible** = Kullback-Leibler (KL) divergence (typically)
- choose the setting for the variational parameters λ that brings $q(y|\lambda)$ closest to $p(y|x)$

3. Inference engine (4)

What distributions can we make use? Graphical model as exponential family.

- Having a set of independent and identically distributed observations of a random variable (i.e., a node in the graph) – many distributions consistent with the observations – we choose the distribution with maximum Shannon entropy (the distribution in exponential family form).
- The exponential family is a parameterized family of distributions, all sharing a similar functional form, and differing only in choice of particular parameters.



Mean-field assumption

$$q(y) = \prod q(y_i)$$

3.Inference engine (5)

ELBO
(Evidence Lower BOund)
(easy to evaluate for given q)

$$log\ p(x) = KL(q\,||\,p) + \mathcal{L}(q)$$

constant w.r.t. q

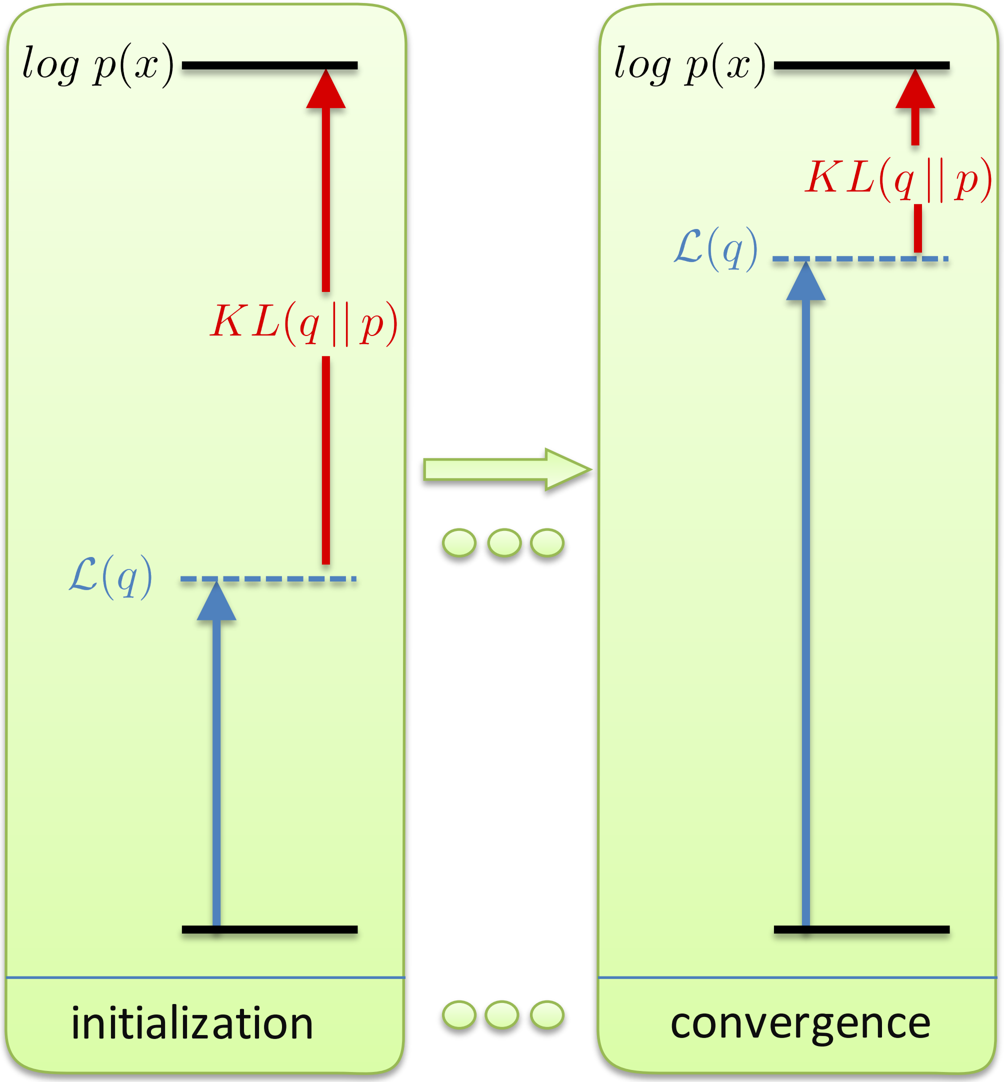
Kullback-Leibler divergence from the variational distribution to the posterior distribution **(unknown and >= 0)**

The ELBO can be rewritten without any reference to the posterior or the marginal distribution.

$$\mathcal{L}(q) = \underbrace{\mathbb{E}_q\left[log\, p(x,y)\right]}_{\text{Expected log joint}} - \underbrace{\mathbb{E}_q\left[log\, q(y)\right]}_{\text{Entropy of the variational distribution q}}$$

objective function

where $\mathbb{E}_f(g(u)) = \int g(u) f(u) du$



Goal

Maximize the objective function $\mathcal{L}(q)$ w.r.t. the variational parameters λ

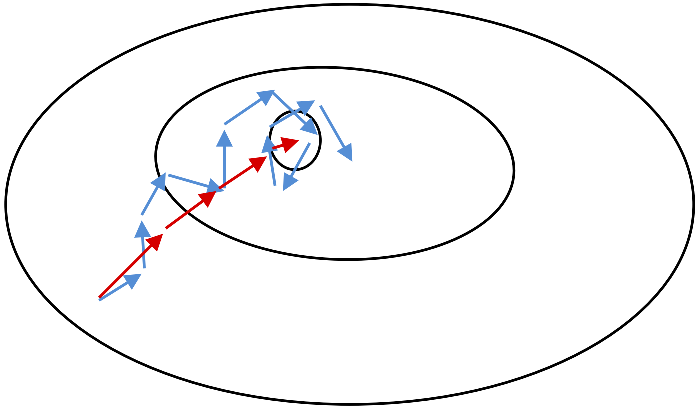
3.Inference engine (6)

Goal

Maximize the objective function $\mathcal{L}(\lambda)$.

The most straightforward way is by using gradient descent.

Stochastic gradient optimization is among the most effective algorithms as concerns the "predictive accuracy obtained per unit of computation". [1]



Gradient descent:

$$\lambda^{(t)} = \lambda^{(t-1)} + \rho_t \cdot \nabla\mathcal{L}(\lambda)$$

$\nabla\mathcal{L}(\lambda)$ depends on $\sum x_i$

- Follow the true gradient
- Expensive to compute

Stochastic gradient descent:

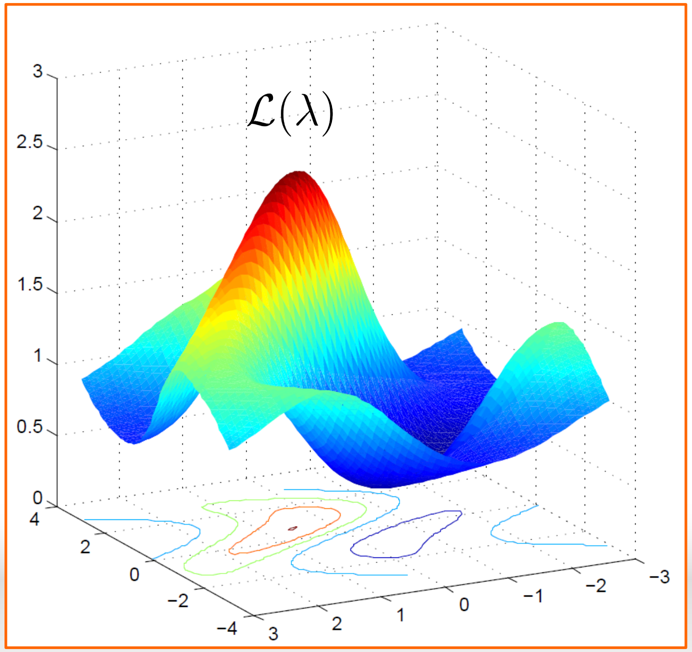
$$\lambda^{(t)} = \lambda^{(t-1)} + \rho_t \cdot \hat{\nabla}\mathcal{L}(\lambda)$$

$\hat{\nabla}\mathcal{L}(\lambda) = \nabla\mathcal{L}_i(\lambda)$ depends solely on x_i

- Follow noisy estimates of the gradient with a decreasing step size
- Fast; allows us to scale to large networks

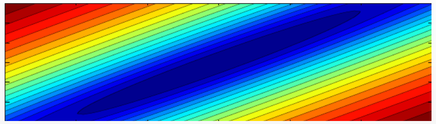
3.Inference engine (7)

e.g.:



Challenges:

- Complex functional landscapes
  - Local saddle points, optima, etc.
  - Highly non-isotropic local behavior



  - Correlation between all dimensions
- High dimenstionality, e.g., hundreds
- Costly evaluation of $\mathcal{L}(\lambda)$

The natural gradient [2]:

- Fast isotropic convergence
- Very efficient in any space - independent of the model parametrization and of the dependencies among signals
  - Invariant w.r.t. change of coordinates λ
  - Invariant to variable transformations y

3.Inference engine (8)

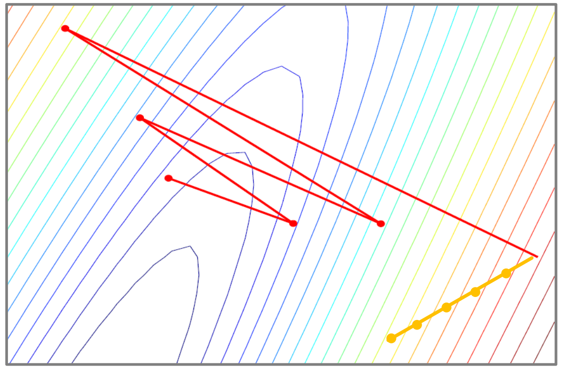
The standard gradient: ∇

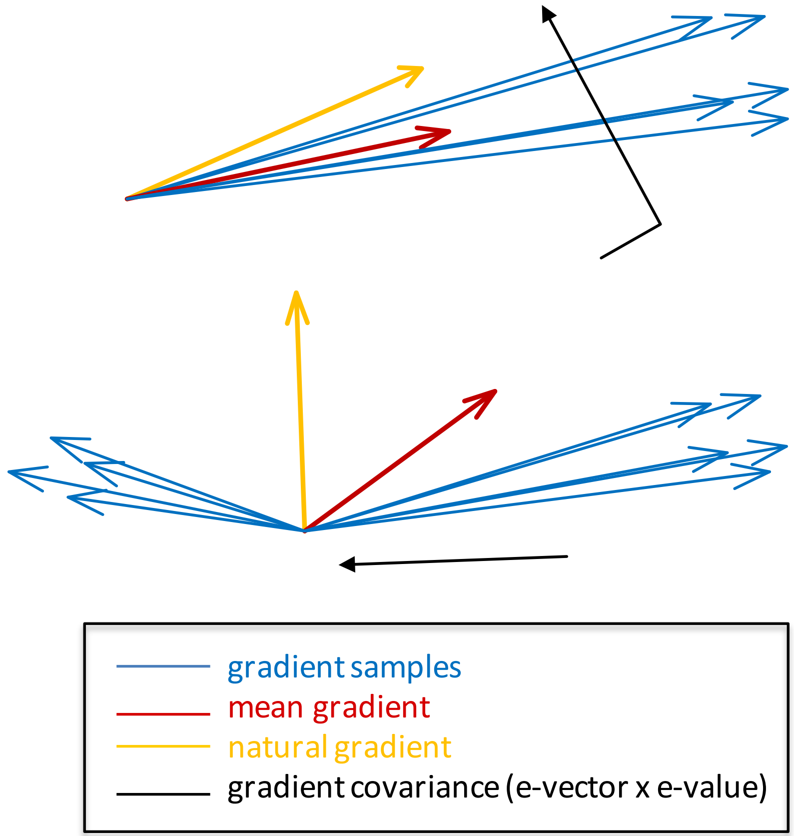
- The normal gradient doesn't work:
  - Over-aggressive steps on ridges;
  - Too small steps on plateaus;
  - Slow or premature convergence, non-robust performance.

The natural gradient: $\tilde{\nabla}$

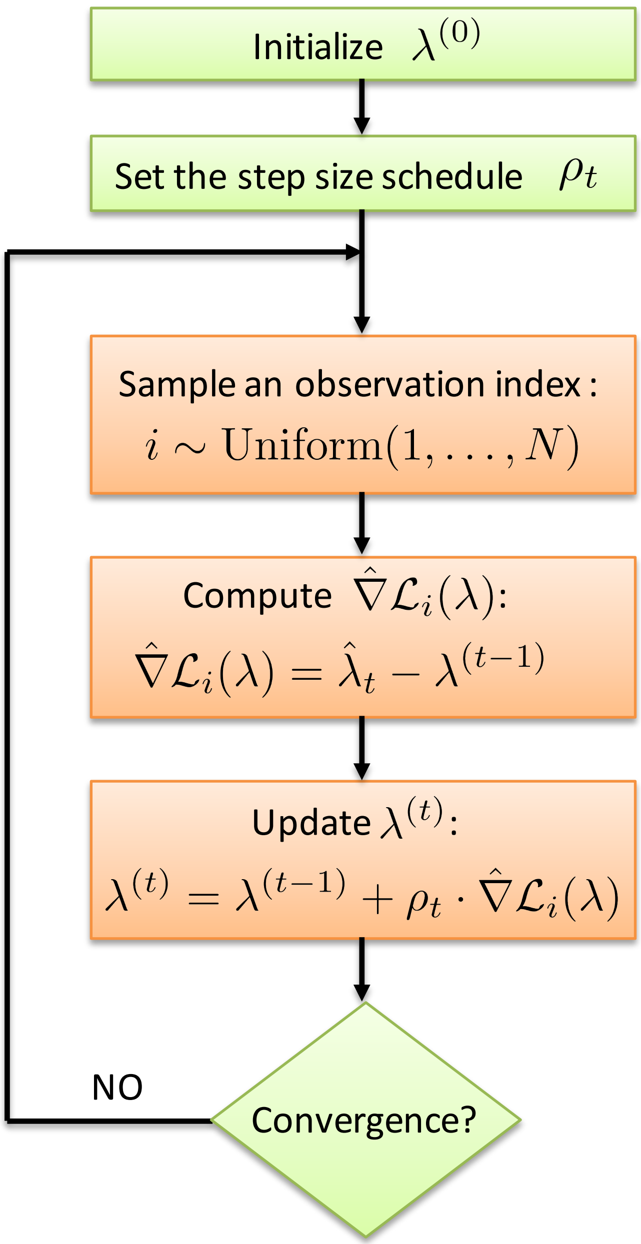
$\tilde{\nabla} = -C^{-1} \cdot \nabla$ with C the covariance of the gradients

Follow the direction where gradients agree (less variability in the data).





3.Inference engine – putting it all together (1)



Initialize randomly the variational parameters.

Choosing the sequence of step sizes can be difficult:

- If it decays too quickly => long time to converge;
- If it decays too slowly => λ will oscillate too much.

Choose an index of the observation data, uniformly at random.

Based on the current sample x_i, compute the noisy (but unbiased) natural gradient of $\mathcal{L}_i$.

Set the new estimate of the variational parameter to be a weighted average of the previous estimate and the current noisy gradient.

3.Inference engine – putting it all together (2)

Initialize $\lambda^{(0)}$

↓

Set the step size schedule ρ_t

Initialize randomly the variational parameters.

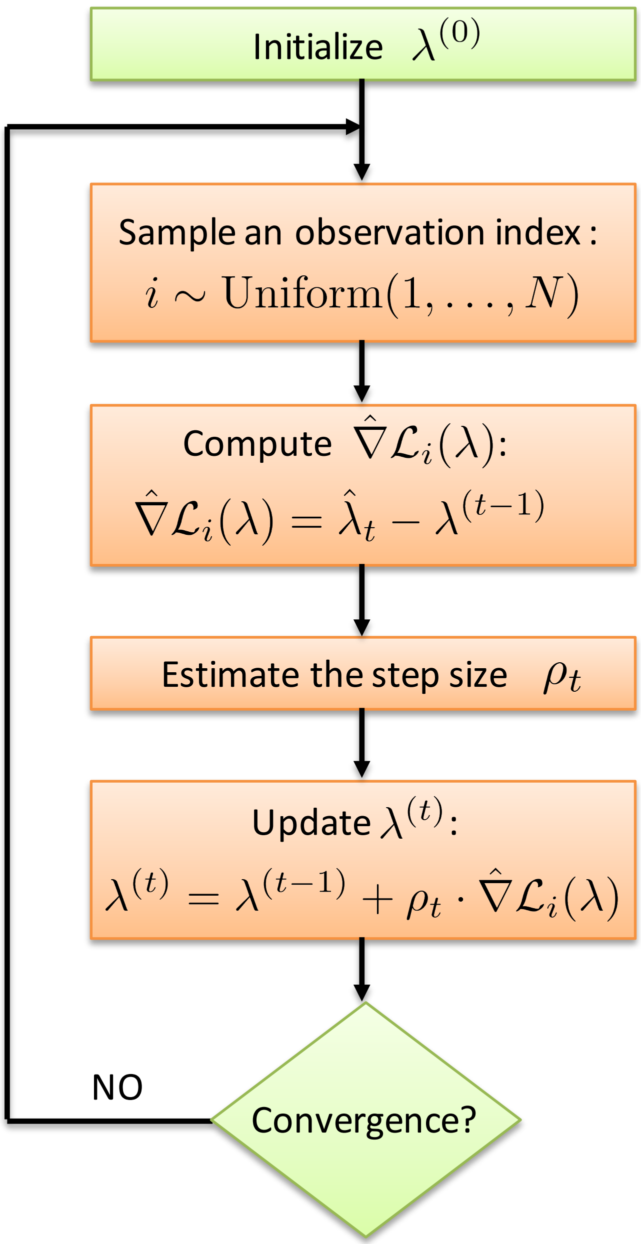
Choosing the sequence of step sizes can be difficult:

- If it decays too quickly => long time to converge;
- If it decays too slowly => λ will oscillate too much.

Solution? – adaptive step size

- Adapt the step size according to the current observation sample x_i
  Specifically: minimize the expected distance between the current stochastic update $\lambda^{(t)}$ and the optimal batch update (when processing the entire dataset x). [4]

3.Inference engine – putting it all together (3)



Initialize randomly the variational parameters.

Choose an index of the observation data, uniformly at random.

Based on the current sample x_i, compute the noisy (but unbiased) natural gradient of $\mathcal{L}_i$.

Estimate the step size for the current stochastic update.

Set the new estimate of the variational parameter to be a weighted average of the previous estimate and the current noisy gradient.

4. Summary and future developments

Summary:

Hierarchical reliability assessment framework

Gate-level – more accurate – Monte Carlo estimates

Circuit-level – probability inference

Variational inference to cope with the dimensionality/precision specific of large circuits.

Future work:

Update the current framework to derive the marginal probability (the probability of a subset of nodes).

References

[1] – L. Bottou and O. Bousquet. **The tradeoffs of large scale learning.** In Advances in Neural Information Processing Systems, pp. 161-168, 2008
[2] – S. Amari. **Natural gradient works efficiently in learning.** Neural Computation, pp. 251-276, 1998
[3] – H, Robbins and S. Monro. **A stochastic approximation method.** Annals of Mathematical Statistics, pp. 404-407, 1051
[4] – T. SchaulS. Zhang, and Y. LeCun. **No more pesky learning rates.** ArXiv e-prints, 2012
[5] – M. Stanisavljevic, A. Schmid, and Y. Leblebici - **Reliability of nanoscale circuits and systems.** Springer, 2011
[6] – Y. Bengio - **Speeding up stochastic gradient descent.** NIPS workshop on efficient machine learning, 2007
[7] – Y. Sun - **Stochastic search using the natural gradients.** 26th International conference on machine learning, 2009
[8] – http://www.fil.ion.ucl.ac.uk/~jdaunize/presentations/Bayes2.pdf
[9] - http://www.eetimes.com/electronics-news/4374306/Imec-looks-at-variability-issue-beyond-10nm-
[10] – F.S. Torres, R.P. Bastos – **Robust modular bulk built-in current sensors for detection of transient faults.** 25th Symposium on integrated circuits and systems design, pp. 1-6, 2012

Thank you!