Lecture 16

Nonlinear Problems:

Simulated Annealing and Bootstrap Confidence Intervals

Syllabus

Lecture 01	Describing Inverse Problems
Lecture 02	Probability and Measurement Error, Part 1
Lecture 03	Probability and Measurement Error, Part 2
Lecture 04	The L_2 Norm and Simple Least Squares
Lecture 05	A Priori Information and Weighted Least Squared
Lecture 06	Resolution and Generalized Inverses
Lecture 07	Backus-Gilbert Inverse and the Trade Off of Resolution and Variance
Lecture 08	The Principle of Maximum Likelihood
Lecture 09	Inexact Theories
Lecture 10	Nonuniqueness and Localized Averages
Lecture 11	Vector Spaces and Singular Value Decomposition
Lecture 12	Equality and Inequality Constraints
Lecture 13	L_1 , L_∞ Norm Problems and Linear Programming
Lecture 14	Nonlinear Problems: Grid and Monte Carlo Searches
Lecture 15	Nonlinear Problems: Newton's Method
Lecture 16	**Nonlinear Problems: Simulated Annealing and Bootstrap Confidence Intervals**
Lecture 17	Factor Analysis
Lecture 18	Varimax Factors, Empircal Orthogonal Functions
Lecture 19	Backus-Gilbert Theory for Continuous Problems; Radon's Problem
Lecture 20	Linear Operators and Their Adjoints
Lecture 21	Fréchet Derivatives
Lecture 22	Exemplary Inverse Problems, incl. Filter Design
Lecture 23	Exemplary Inverse Problems, incl. Earthquake Location
Lecture 24	Exemplary Inverse Problems, incl. Vibrational Problems

Purpose of the Lecture

Introduce Simulated Annealing

Introduce the Bootstrap Method for computing Confidence Intervals

Part 1

Simulated Annealing

Monte Carlo Method
completely undirected

Newton's Method
completely directed

Monte Carlo Method
completely undirected

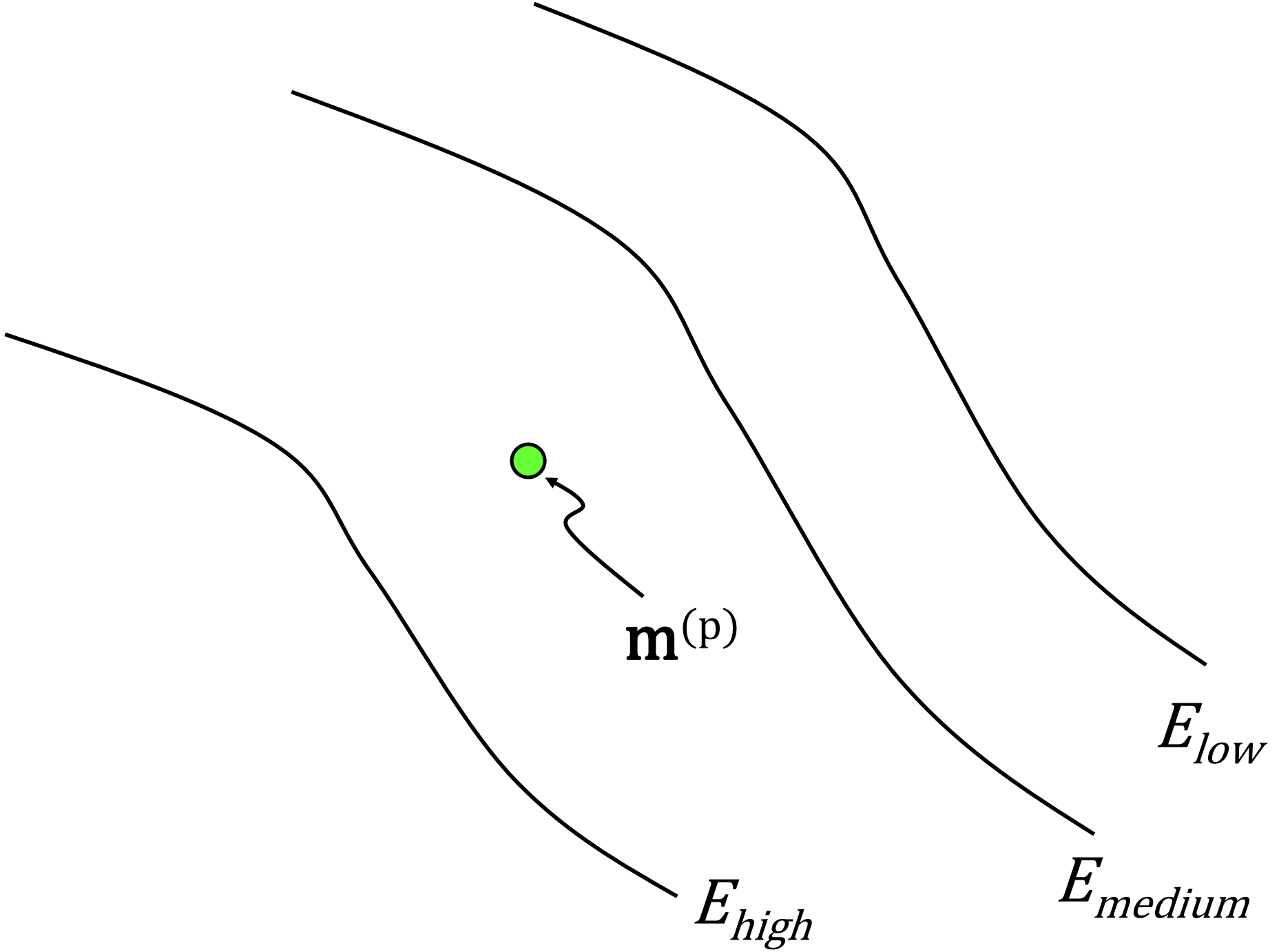
slow, but
foolproof

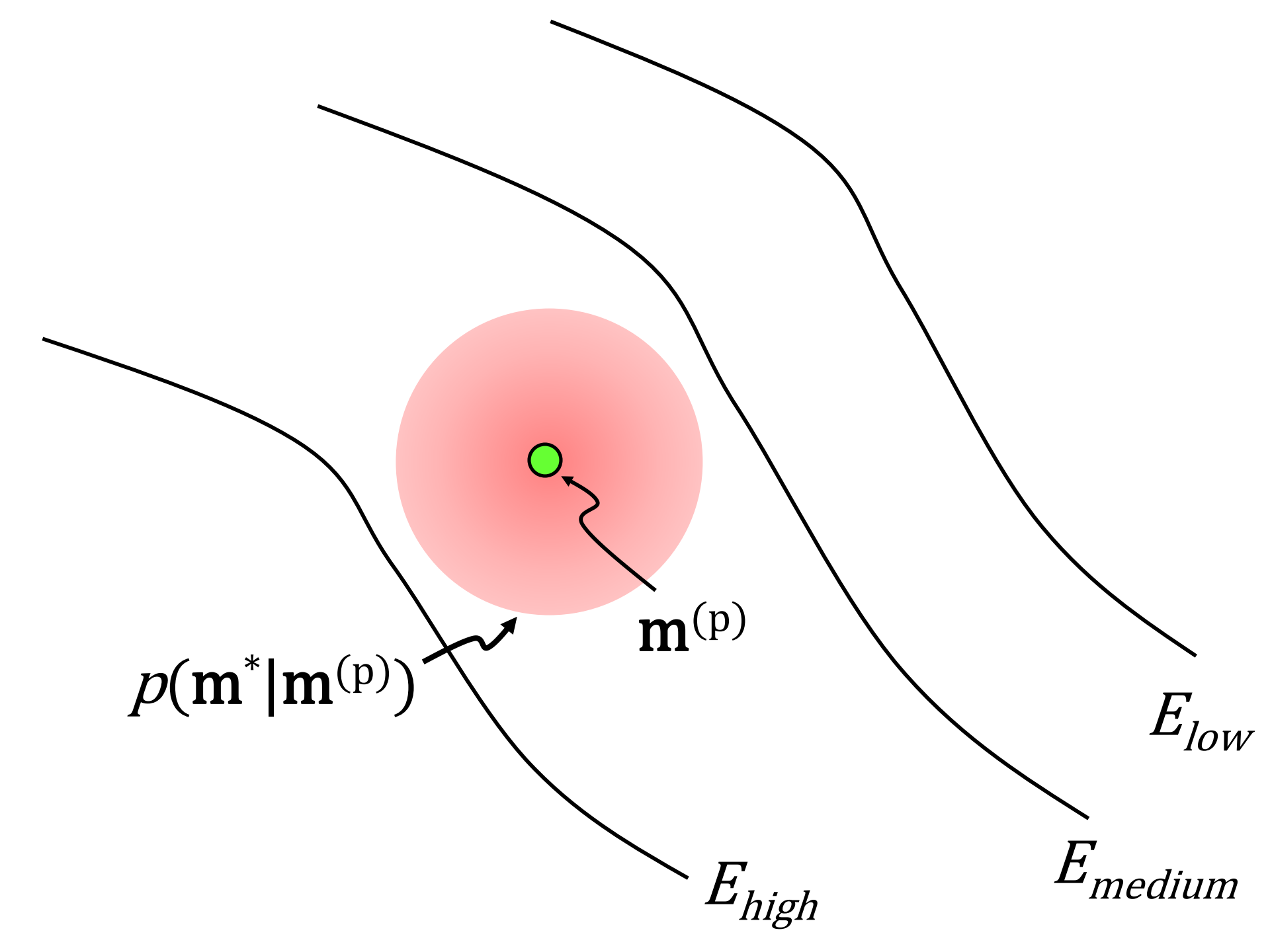
Newton's Method
completely directed

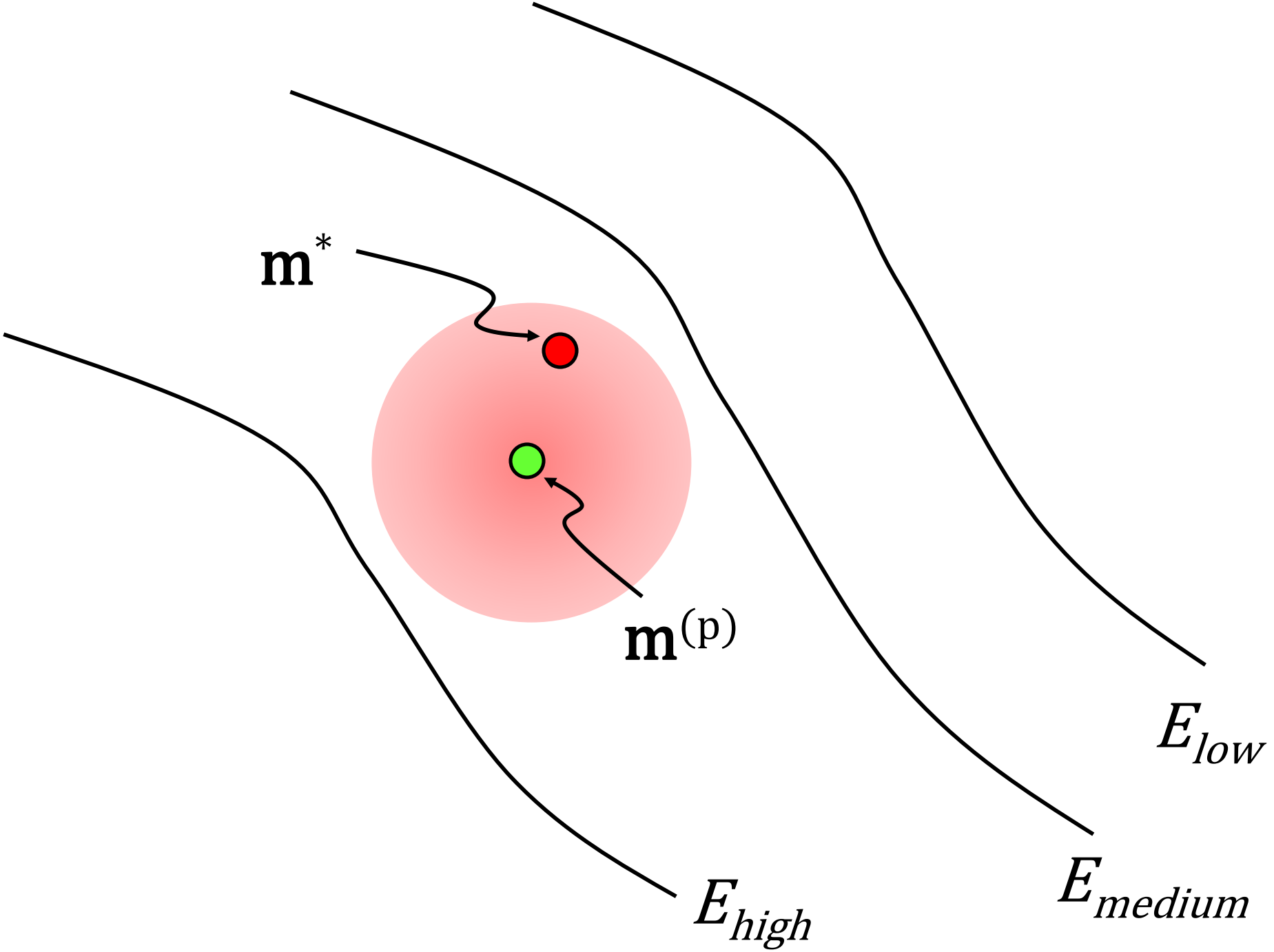
fast, but can fall
into local minimum

compromise

partially-directed random walk


$\mathbf{m}^{(p)}$
E_{low}
E_{medium}
E_{high}


$p(\mathbf{m}^*|\mathbf{m}^{(p)})$
$\mathbf{m}^{(p)}$
E_{low}
E_{medium}
E_{high}


$\mathbf{m}^*$
$\mathbf{m}^{(p)}$
E_{low}
E_{medium}
E_{high}

acceptance of $\mathbf{m}^*$ as $\mathbf{m}^{(p+1)}$

always accept in error is smaller

accept with probability

$$\exp\left\{-\frac{\left[E(\mathbf{m}^*) - E\left(\mathbf{m}^{(p)}\right)\right]}{T}\right\}$$

where T is a parameter
if error is bigger

large T

$$\exp\left\{-\frac{\left[E(\mathbf{m}^*) - E\left(\mathbf{m}^{(p)}\right)\right]}{T}\right\} \longrightarrow 1$$

always accept $\mathbf{m}^*$
(undirected random walk)
ignores the error completely

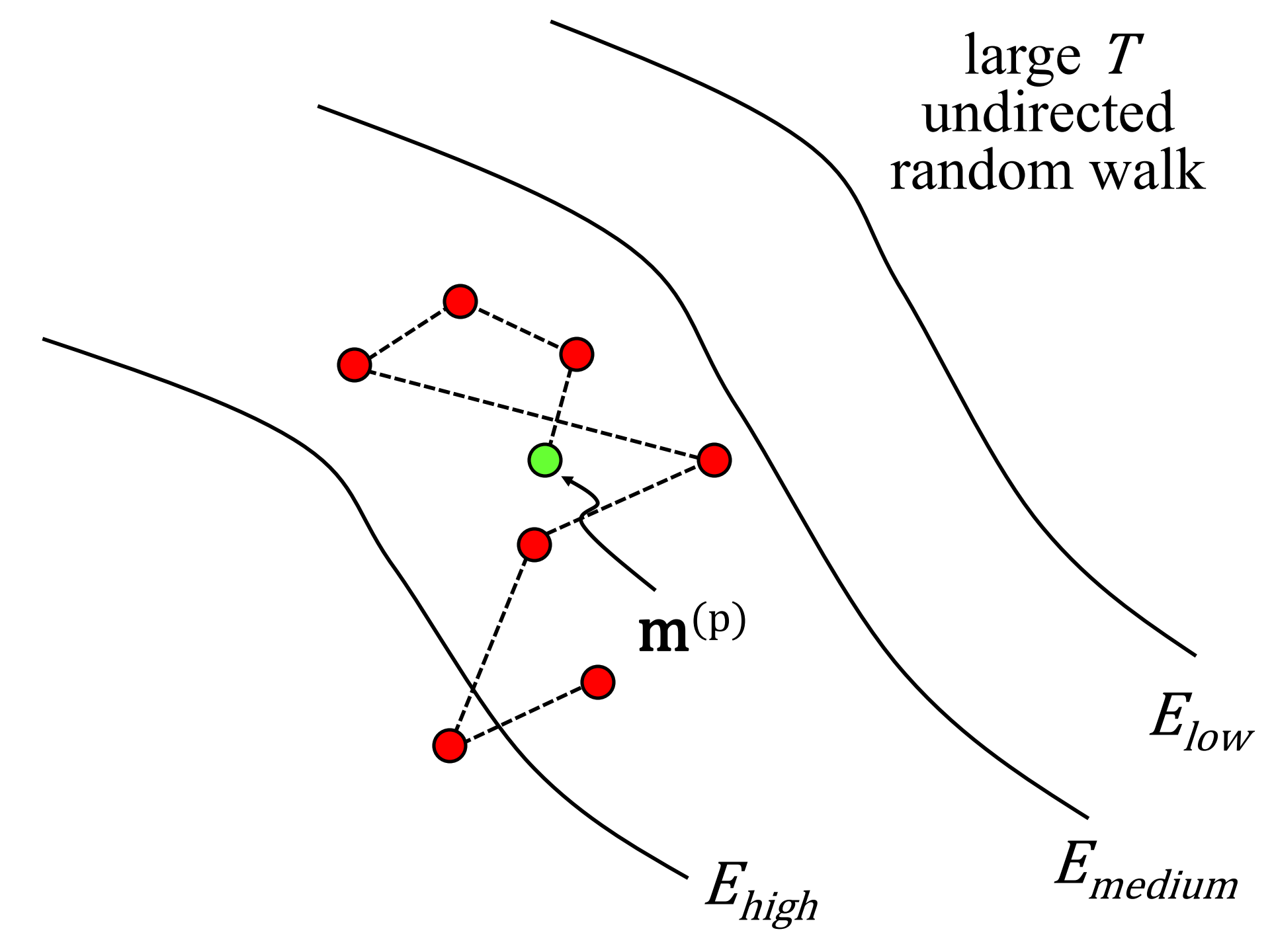
small T

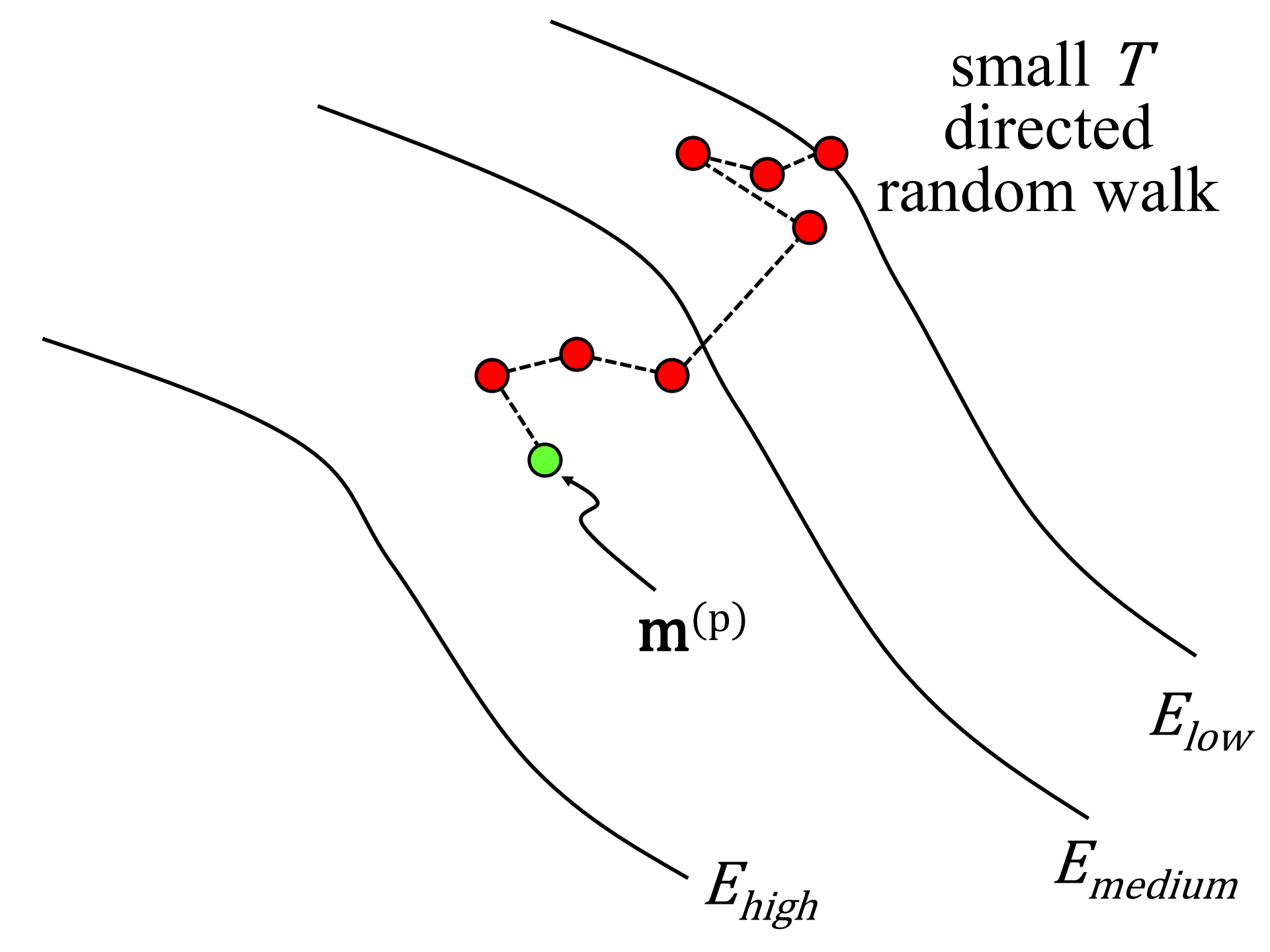
$$\exp\left\{-\frac{\left[E(\mathbf{m}^*) - E\left(\mathbf{m}^{(p)}\right)\right]}{T}\right\} \longrightarrow 0$$

accept $\mathbf{m}^*$ only when error is smaller
(directed random walk)
strictly decreases the error

intermediate T

most iterations decrease the error
but occasionally allow an $\mathbf{m}^*$
that increases it


large T
undirected
random walk
$\mathbf{m}^{(p)}$
E_{low}
E_{medium}
E_{high}


small T
directed
random walk
$\mathbf{m}^{(p)}$
E_{low}
E_{medium}
E_{high}

strategy

start off with large T

undirected
similar to Monte Carlo method
(except more “local”)

slowly decrease T during iterations

directed
similar to Newton’s method
(except precise gradient direction not used)

strategy

start off with large T
more random

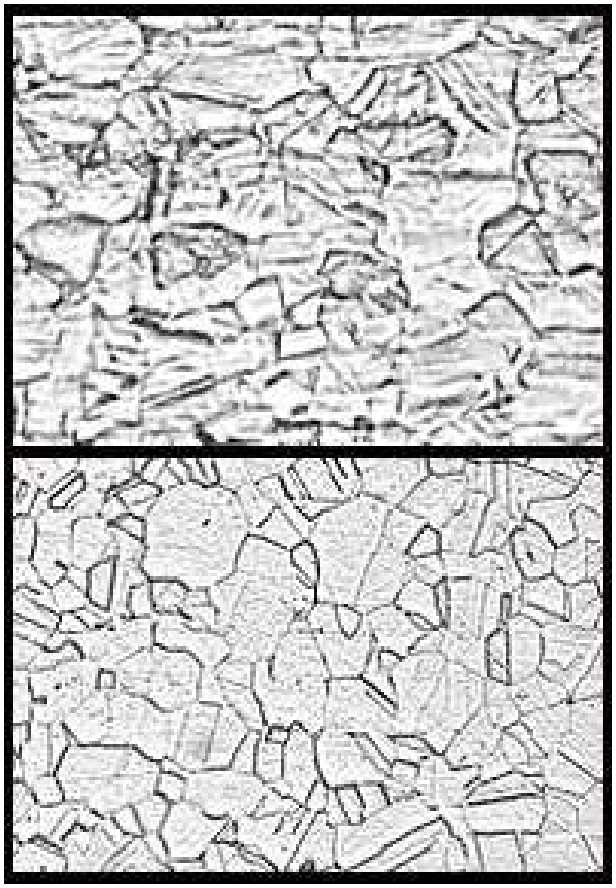
slowly decrease T during iterations
more directed

claim is that this strategy helps achieve the global minimum

analogous to annealing of metals

high temperatures
atoms randomly moving
about due to thermal motions

as temperature decreases
atoms slowly find themselves in a
minimum energy configuration
orderly arrangement of a “crystal”



www.sti-laser.com/technology/heat_treatments.html

analogous to annealing of metals

high temperatures
atoms randomly moving
about due to thermal motions

as temperature decreases
atoms slowly find themselves in a
minimum energy configuration
orderly arrangement of a “crystal”

hence “simulated annealing”
and T called “temperature”

this is just Metroplois-Hastings

(way of producing realizations of a random variable)

applied to the p.d.f.

$$p(\mathbf{m}) \propto \exp\left\{\frac{-E(\mathbf{m})}{T}\right\}$$

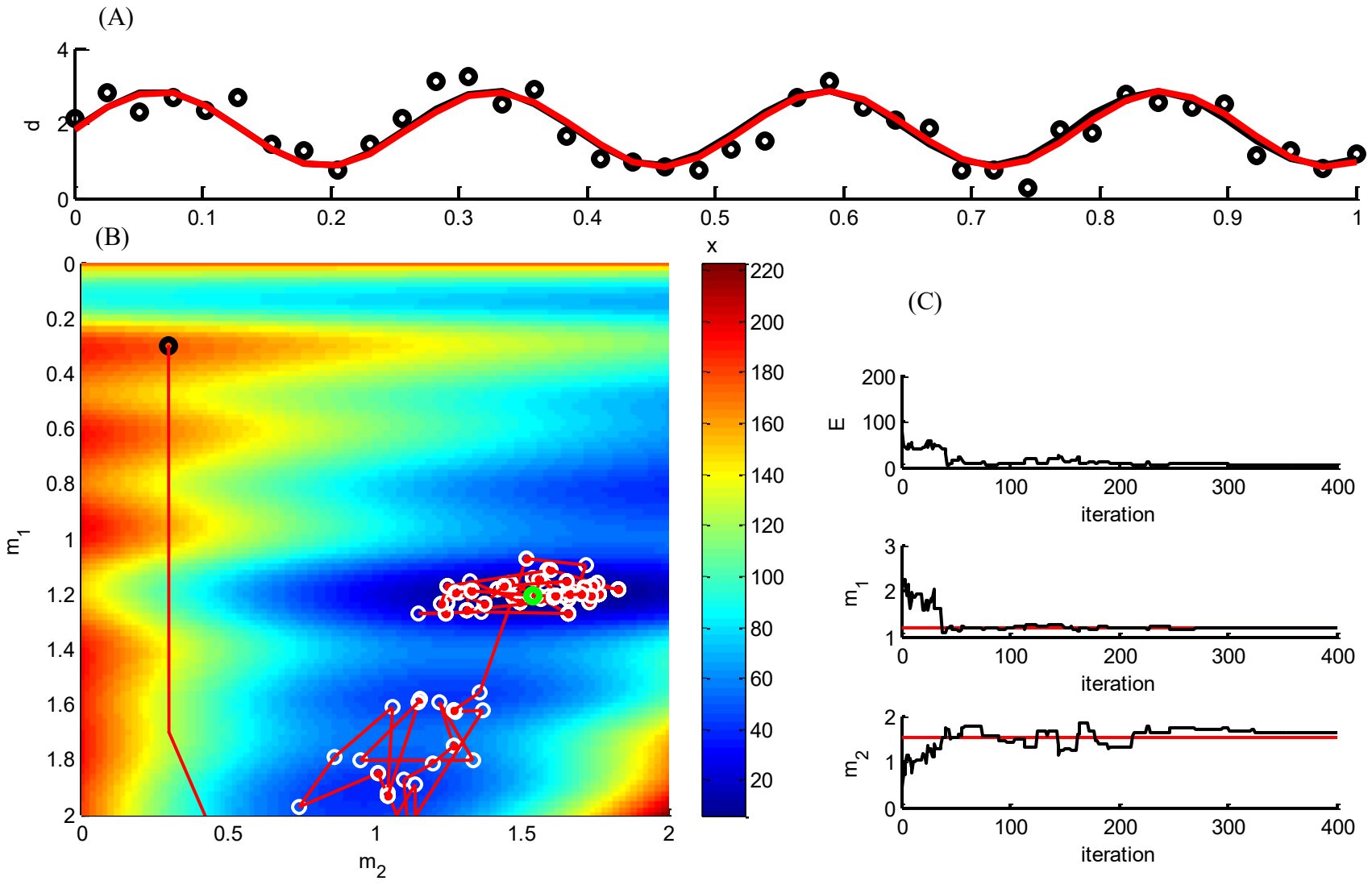
this is just Metroplois-Hastings

(way of producing realizations of a random variable)

applied to the p.d.f.

$$p(\mathbf{m}) \propto \exp\left\{\frac{-E(\mathbf{m})}{T}\right\}$$

sampling a distribution that starts out wide and blurry but sharpens up as T is decreases


(A)
d
x
(B)
m_1
m_2
(C)
E
iteration
m_1
iteration
m_2
iteration

```
for k = [1:Niter]
    T = 0.1 * Eg0 * ((Niter-k+1)/Niter)^2;

    ma(1) = random('Normal',mg(1),Dm);
    ma(2) = random('Normal',mg(2),Dm);
    da = sin(w0*ma(1)*x) + ma(1)*ma(2);
    Ea = (dobs-da)'*(dobs-da);

    if( Ea < Eg )
        mg=ma;
        Eg=Ea;
        p1his(k+1)=1;
    else
        p1 = exp( -(Ea-Eg)/T );
        p2 = random('unif',0,1);
        if( p1 > p2 )
            mg=ma;
            Eg=Ea;
        end
    end
```

Part 2

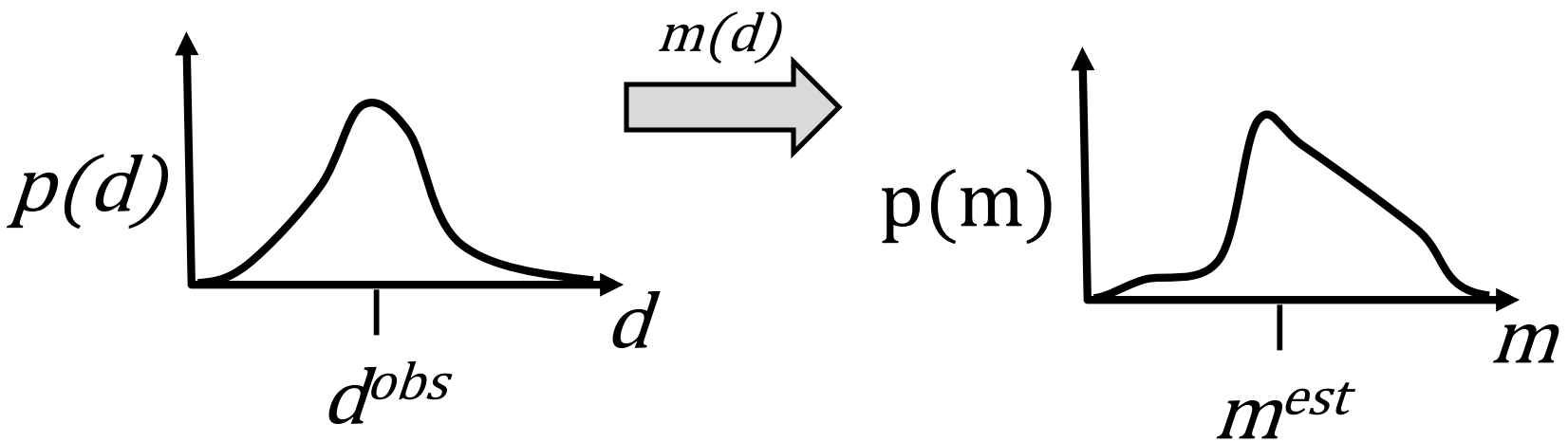
Bootstrap Method

theory of confidence intervals

error is the data

result in

errors in the estimated model parameters

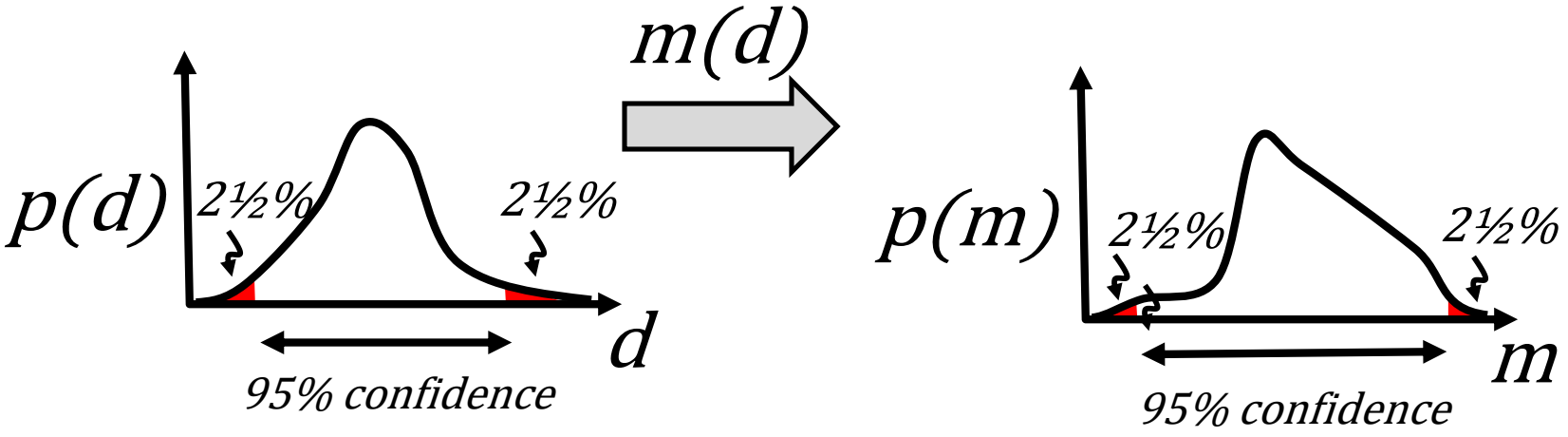


theory of confidence intervals

error is the data

result in

errors in the estimated model parameters



Gaussian linear theory

$$\mathbf{d} = \mathbf{G}\mathbf{m}$$

$$\mathbf{m} = \mathbf{G}^{-g}\mathbf{d}$$

standard error propagation

$$[\mathrm{cov}\ \mathbf{m}] = \mathbf{G}^{-g}\,[\mathrm{cov}\ \mathbf{d}]\,\mathbf{G}^{-gT}$$

univariate Gaussian distribution has 95% of error within two σ of its mean

What to do with Gaussian nonlinear theory?

One possibility
linearize theory and use standard error propagation

$$\mathbf{d} = \mathbf{g}(\mathbf{m})$$

$$\mathbf{m}\text{-}\mathbf{m}^{(p)} \approx \mathbf{G}_{(p)}^{-g}\left[\mathbf{d}\text{-}\,\mathbf{g}(\mathbf{m}^{(p)})\right]$$

$$[\mathrm{cov}\,\mathbf{m}] \approx \mathbf{G}_{(p)}^{-g}\,[\mathrm{cov}\,\mathbf{d}]\,\mathbf{G}_{(p)}^{-g}$$

disadvantages
unknown accuracy
and
need to compute gradient of theory $\mathbf{G}_{(p)}$

$\mathbf{G}_{(p)}$ not computed when using some solution methods

alternative confidence intervals with repeat datasets

do the whole experiment many times

use results of each experiment to make compute $\mathbf{m}^{est}$

create histograms from many $\mathbf{m}^{est}$'s

derive empirical 95% confidence intervals from histograms

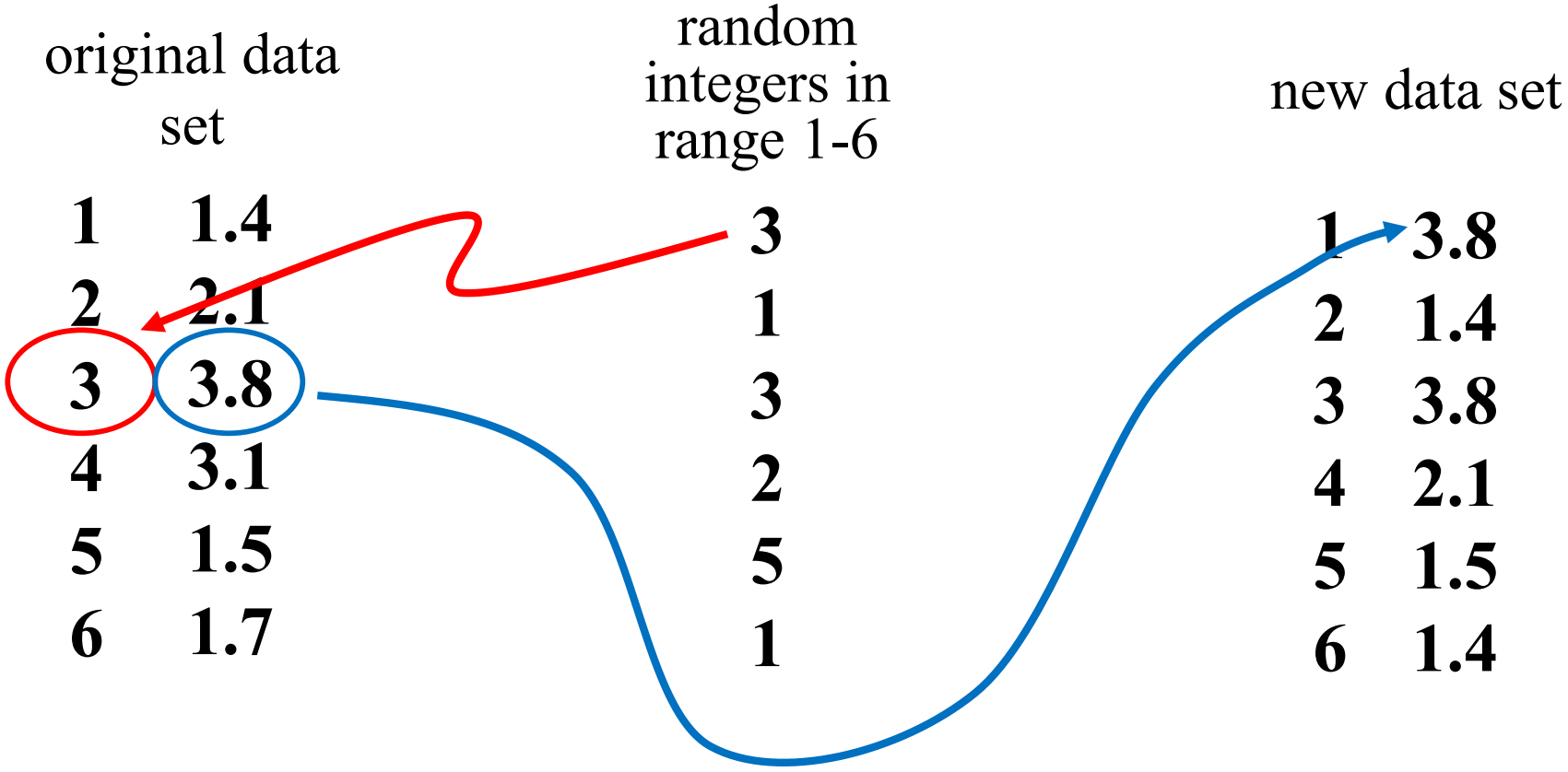
Bootstrap Method

create approximate repeat datasets
by randomly resampling (with duplications)
the one existing data set

example of resampling

original data set		random integers in range 1-6	resampled data set	
1	**1.4**	**3**	**1**	**3.8**
2	**2.1**	**1**	**2**	**1.4**
3	**3.8**	**3**	**3**	**3.8**
4	**3.1**	**2**	**4**	**2.1**
5	**1.5**	**5**	**5**	**1.5**
6	**1.7**	**1**	**6**	**1.4**

example of resampling



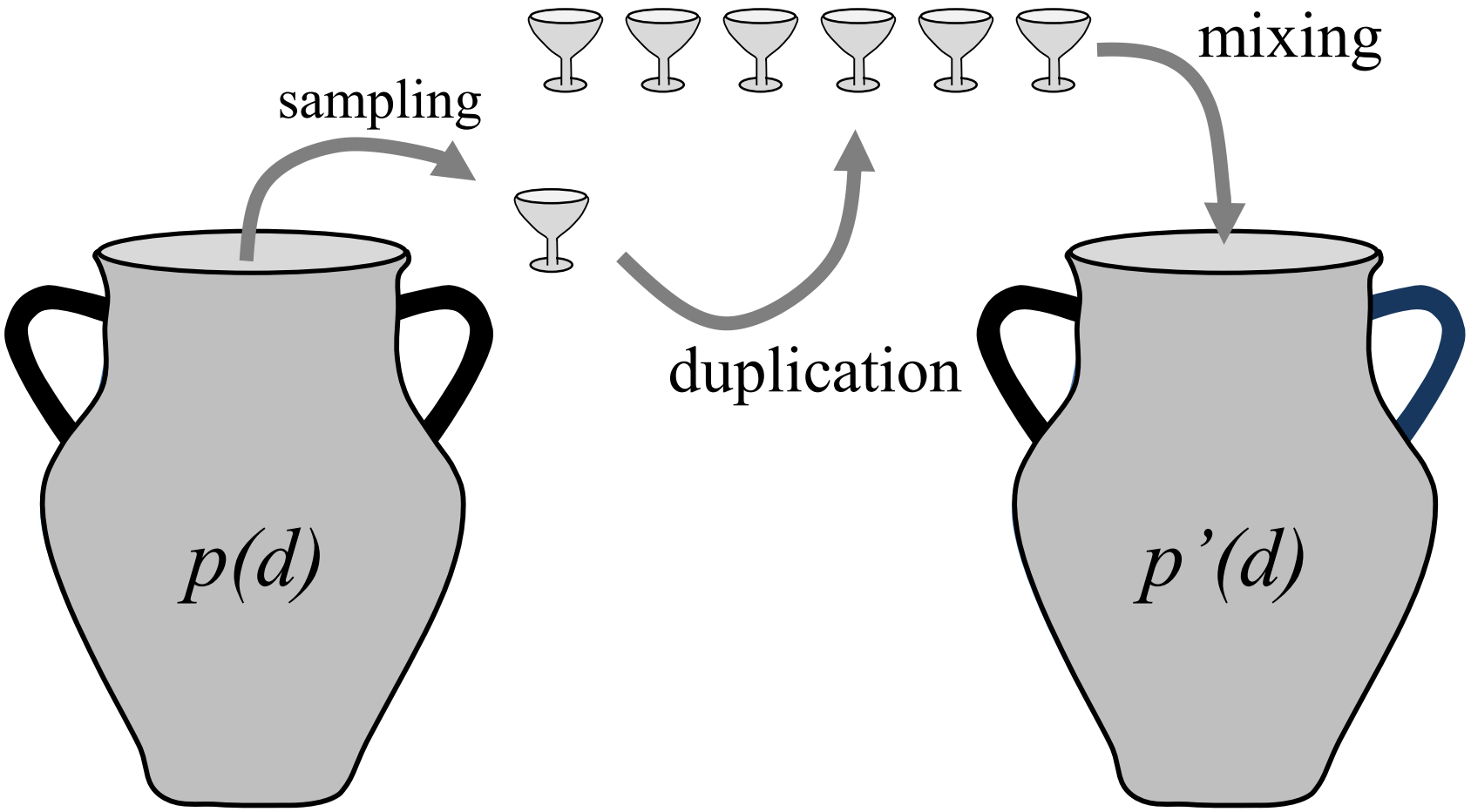
example of resampling

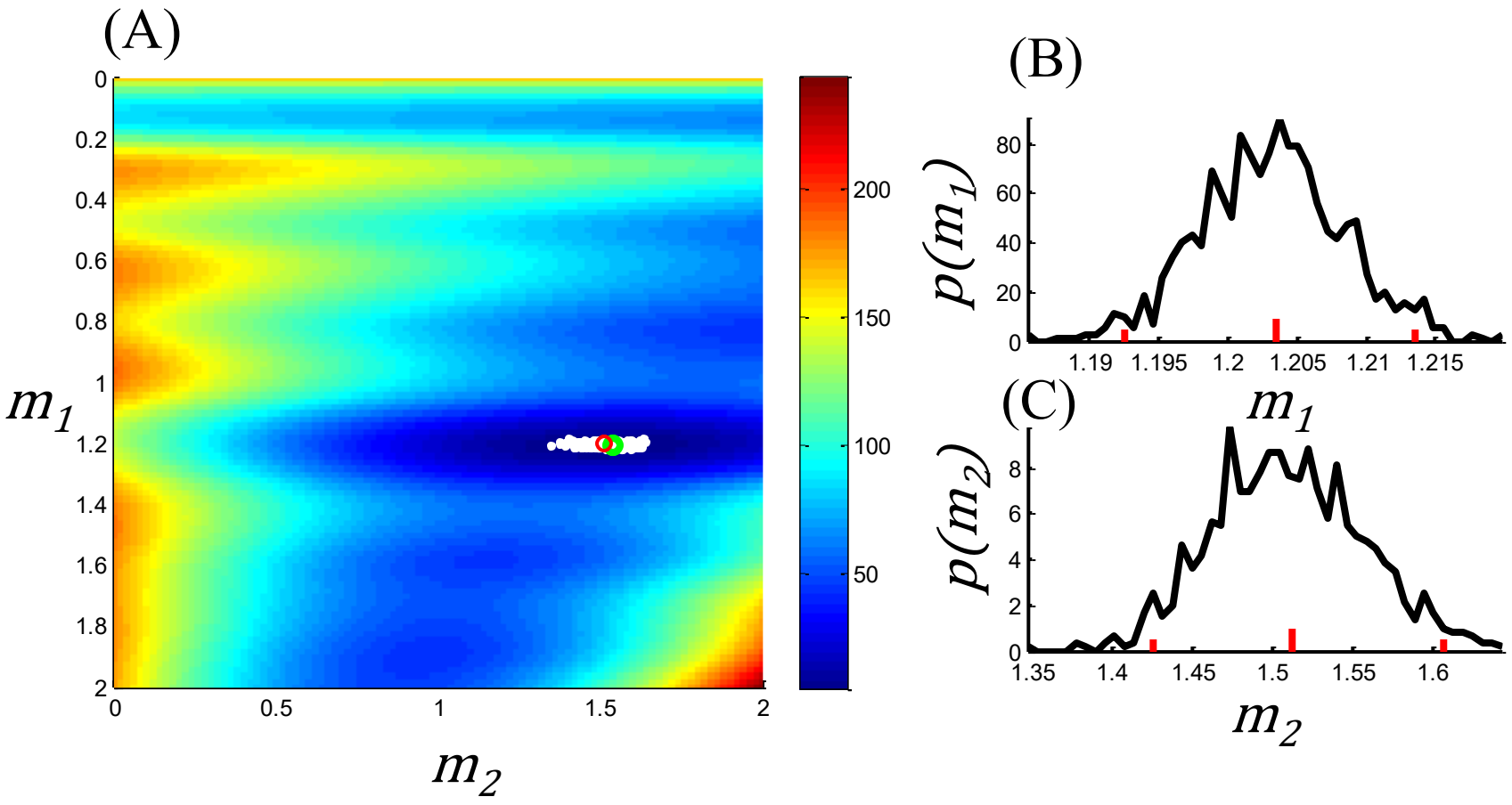
original data set		random integers in range 1-6	resampled data set	
1	**1.4**	**3**	**1**	**3.8**
2	**2.1**	**1**	**2**	**1.4**
3	**3.8**	**3**	**3**	**3.8**
4	**3.1**	**2**	**4**	**2.1**
5	**1.5**	**5**	**5**	**1.5**
6	**1.7**	**1**	**6**	**1.4**

note repeats

```
rowindex = unidrnd(N,N,1);
xresampled = x( rowindex );
dresampled = dobs( rowindex );
```

interpretation of resampling




(A)
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
m_1
0
0.5
1
1.5
2
m_2
200
150
100
50
(B)
$p(m_1)$
80
60
40
20
0
1.19
1.195
1.2
1.205
1.21
1.215
m_1
(C)
$p(m_2)$
8
6
4
2
0
1.35
1.4
1.45
1.5
1.55
1.6
m_2

```
Nbins=50;
m1hmin=min(m1save);
m1hmax=max(m1save);
Dm1bins = (m1hmax-m1hmin)/(Nbins-1);
m1bins=m1hmin+Dm1bins*[0:Nbins-1]';
m1hist = hist(m1save,m1bins);
pm1 = m1hist/(Dm1bins*sum(m1hist));
Pm1 = Dm1bins*cumsum(pm1);
m1low=m1bins(find(Pm1>0.025,1));
m1high=m1bins(find(Pm1>0.975,1));
```

```
Nbins=50;
m1hmin=min(m1save);
m1hmax=max(m1save);
Dm1bins = (m1hmax-m1hmin)/(Nbins-1);
m1bins=m1hmin+Dm1bins*[0:Nbins-1]';
m1hist = hist(m1save,m1bins);
pm1 = m1hist/(Dm1bins*sum(m1hist));
Pm1 = Dm1bins*cumsum(pm1);
m1low=m1bins(find(Pm1>0.025,1));
m1high=m1bins(find(Pm1>0.975,1));
```

histogram

```
Nbins=50;
m1hmin=min(m1save);
m1hmax=max(m1save);
Dm1bins = (m1hmax-m1hmin)/(Nbins-1);
m1bins=m1hmin+Dm1bins*[0:Nbins-1]';
m1hist = hist(m1save,m1bins);
pm1 = m1hist/(Dm1bins*sum(m1hist));
Pm1 = Dm1bins*cumsum(pm1);
m1low=m1bins(find(Pm1>0.025,1));
m1high=m1bins(find(Pm1>0.975,1));
```

empirical p.d.f.

```
Nbins=50;
m1hmin=min(m1save);
m1hmax=max(m1save);
Dm1bins = (m1hmax-m1hmin)/(Nbins-1);
m1bins=m1hmin+Dm1bins*[0:Nbins-1]';
m1hist = hist(m1save,m1bins);
pm1 = m1hist/(Dm1bins*sum(m1hist));
Pm1 = Dm1bins*cumsum(pm1);
m1low=m1bins(find(Pm1>0.025,1));
m1high=m1bins(find(Pm1>0.975,1));
```

empirical c.d.f.

```
Nbins=50;
m1hmin=min(m1save);
m1hmax=max(m1save);
Dm1bins = (m1hmax-m1hmin)/(Nbins-1);
m1bins=m1hmin+Dm1bins*[0:Nbins-1]';
m1hist = hist(m1save,m1bins);
pm1 = m1hist/(Dm1bins*sum(m1hist));
Pm1 = Dm1bins*cumsum(pm1);
m1low=m1bins(find(Pm1>0.025,1));
m1high=m1bins(find(Pm1>0.975,1));
```

95% confidence bounds