Inverse Problems in Geophysics
Part 12

2. MGPY+MGIN

Thomas Günther
thomas.guenther@geophysik.tu-freiberg.de

TU BERGAKADEMIE FREIBERG

TUBAF
Die Ressourcenuniversität.
Seit 1765.

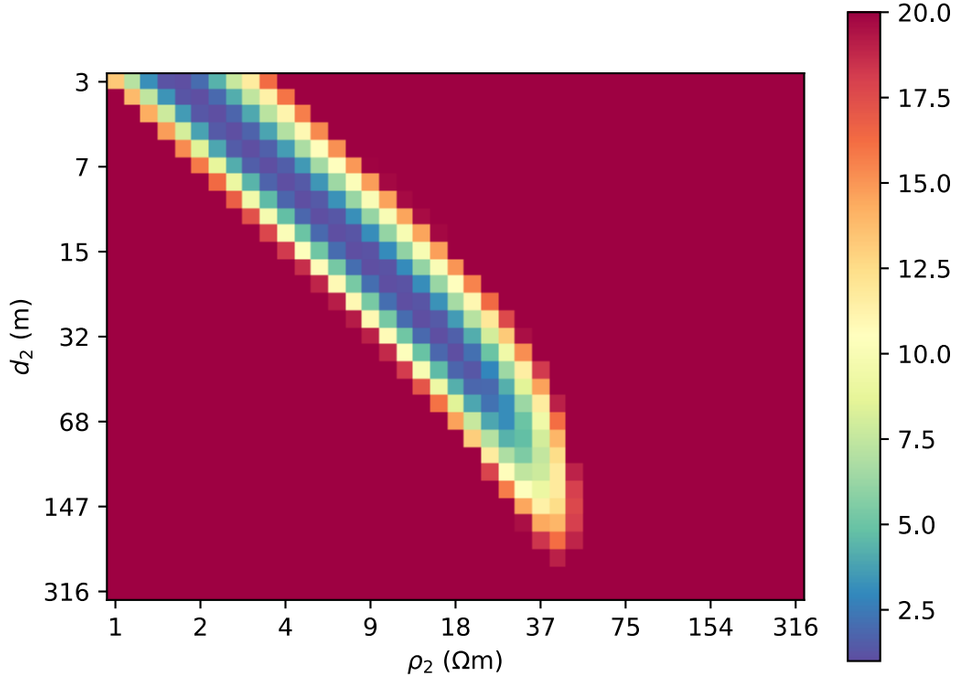
Recap non-linear

undirected search methods

- grid search
- random sampling (Monte Carlo)
- simulated annealing

directed search methods

- gradients (steepest descent)
- Newtons method (linearization)



Objective function for VES

Recap probability

Probability & Likelihood

“probability” und “likelihood” werden beide mit “Wahrscheinlichkeit” übersetzt, haben aber unterschiedliche Bedeutungen. “Probability” bezieht sich auf die Wahrscheinlichkeit eines Ereignisses, das eintreten kann, während “likelihood” die Wahrscheinlichkeit bestimmter Daten unter der Annahme ist, dass ein bestimmtes Modell oder eine bestimmte Hypothese wahr ist

Bedingte Wahrscheinlichkeit

$P(A|B)$ Wahrscheinlichkeit dass A eintritt wenn B eintritt.

Recap Bayes Theorem

$$p(A|B) = p(A,B)/p(B) \Rightarrow p(\mathbf{m}|\mathbf{d})p(\mathbf{d}) = p(\mathbf{d}|\mathbf{m})p(\mathbf{m})$$

Bayes theorem

$$p(\mathbf{m}|\mathbf{d}) = \frac{p(\mathbf{d}|\mathbf{m})p(\mathbf{m})}{p(\mathbf{d})}$$

posterior distribution $\propto$ likelihood x prior distribution

Beispiel Schnelltest

- $P(K)$: Wahrscheinlichkeit dass Person krank ist (1%)
- $P(T)$: Wahrscheinlichkeit dass Test positiv ist
- $P(T|K)$: Wahrscheinlichkeit dass Test bei Krankheit anschlägt (90%)

1000 Patienten (10 krank, 990 gesund)

$\Rightarrow$ 1 falsch negativ (9 korrekt), 99 falsch positiv

$$P(K|T) = \frac{P(T|K)P(K)}{P(T)} = \frac{0.9 * 0.1}{108/1000} = 8.3\%$$

Umkehr-Beispiel

Note

Bjarne hört den Fußball schauenden Tim jubeln.
Wie groß ist die Wahrscheinlichkeit, dass ein Tor gefallen ist?

1. Annahme: In 2% der Zeit(spannen) fällt ein Tor.
2. Annahme: Wenn der Nachbar jubelt, ist es zu 90% ein Tor.
3. Annahme: andere Gründe für Jubel (98%) mit Wahrscheinlichkeit von 1%.

$$P(T|J) = \frac{P(J|T)P(T)}{P(J|T) + P(J|-T)} = \frac{0.02 * 0.9}{0.02 * 0.9 + 0.98 * 0.01} = 64.7\%$$

Application of Bayes rule

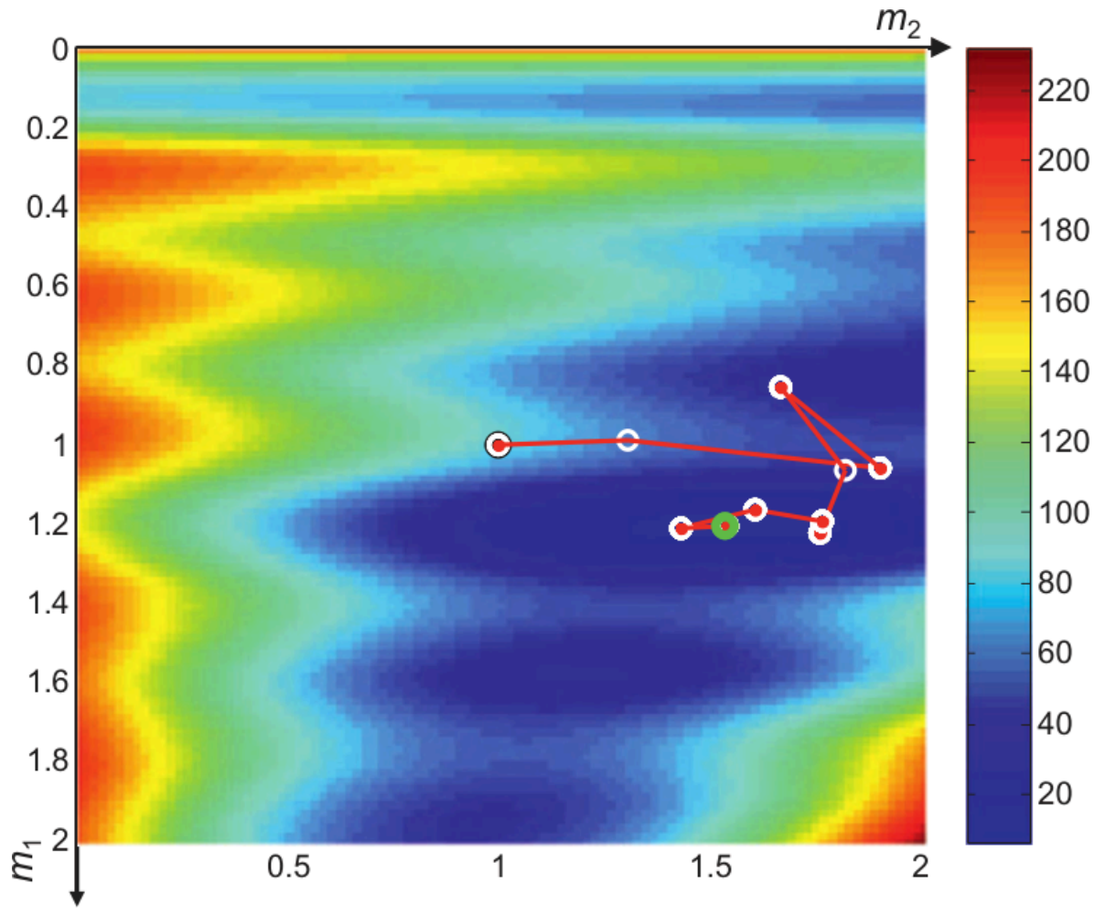
- understanding probability problems (e.g. medicine)
- statistical modelling and inference
- machine learning

Metropolis algorithm

Markow chain

Monte Carlo methods

- Monte Carlos search: randomly draw solutions from grid
- accept solution only if better than old
- Markow-Chain-Monte-Carlo
- Metropolis-Hastings (Metropolis et al., 1953; Hastings, 1970)

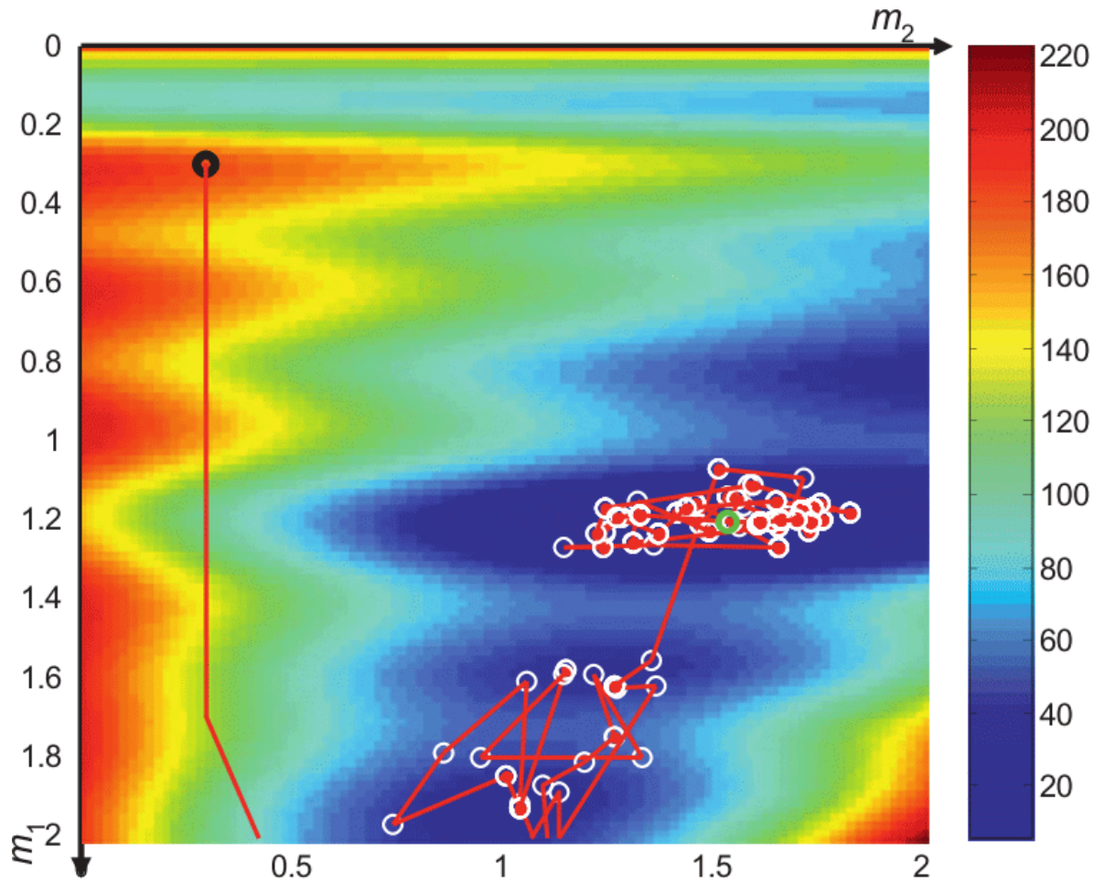


Monte Carlo method

Simulated Annealing

Test parameter

$$P(E) = e^{-(\Phi(\mathbf{m})-\Phi(\mathbf{m}^p))/T}$$



Simulated Annealing

Particle swarm optimization

Alternatives to grid search

Monte Carlo search

draw random samples and accept them if the error is improved

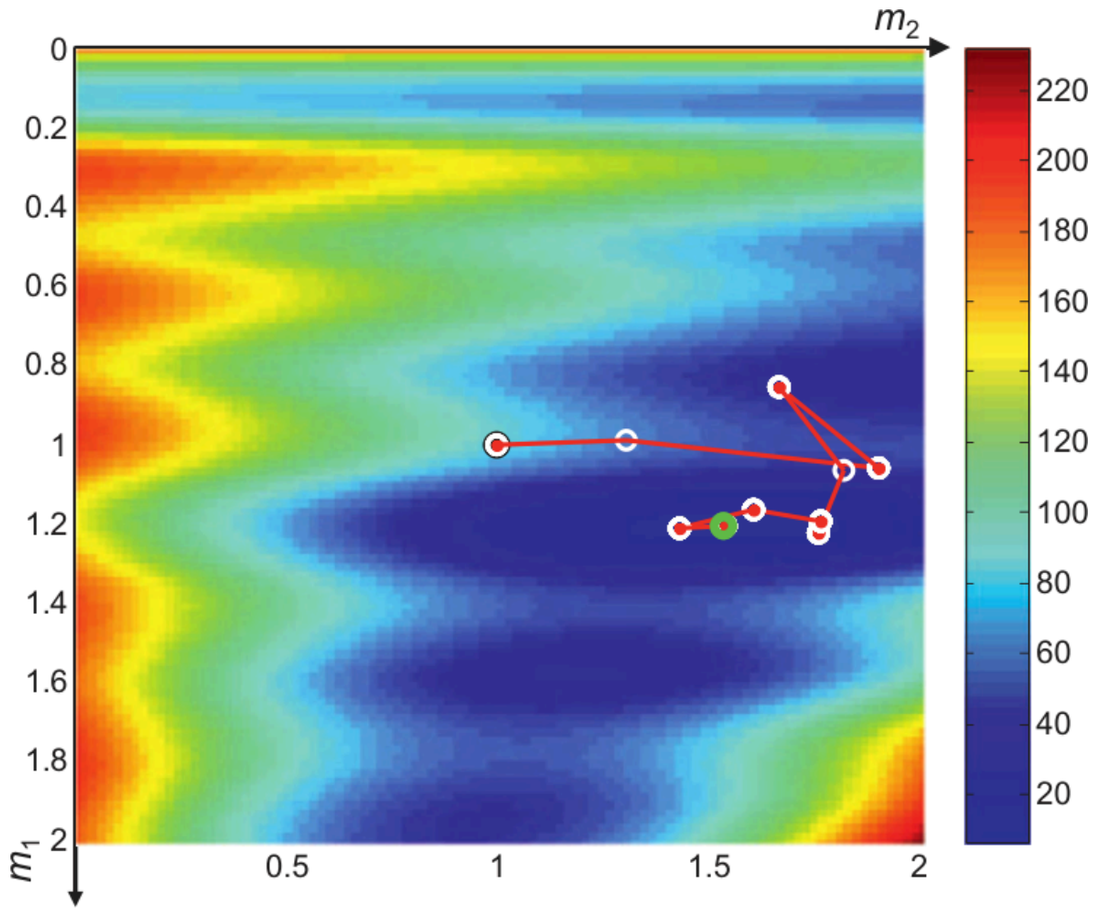
undirected search (Newtons method is directed)

Simulated annealing

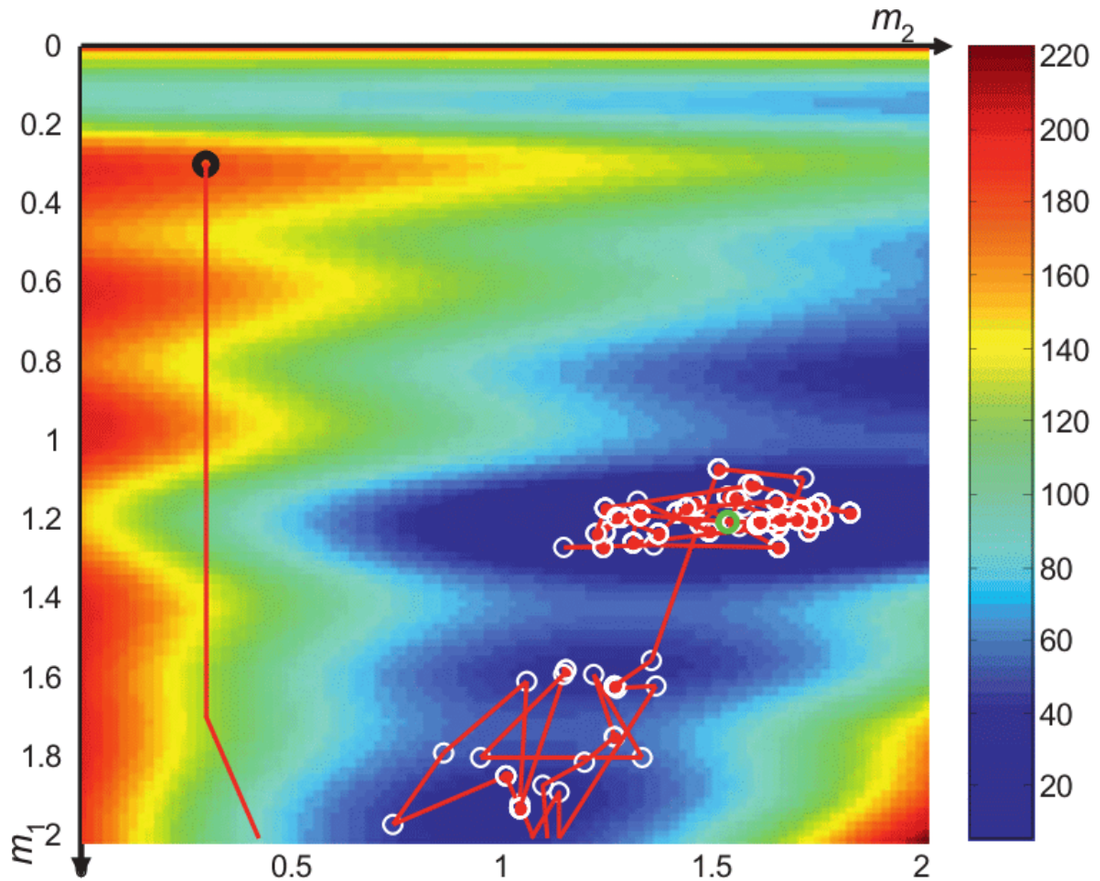
decrease temperature controlling particle movements:

high T: undirected, low T: search in vicinity of current model

Monte Carlo vs. Simulated Annealing

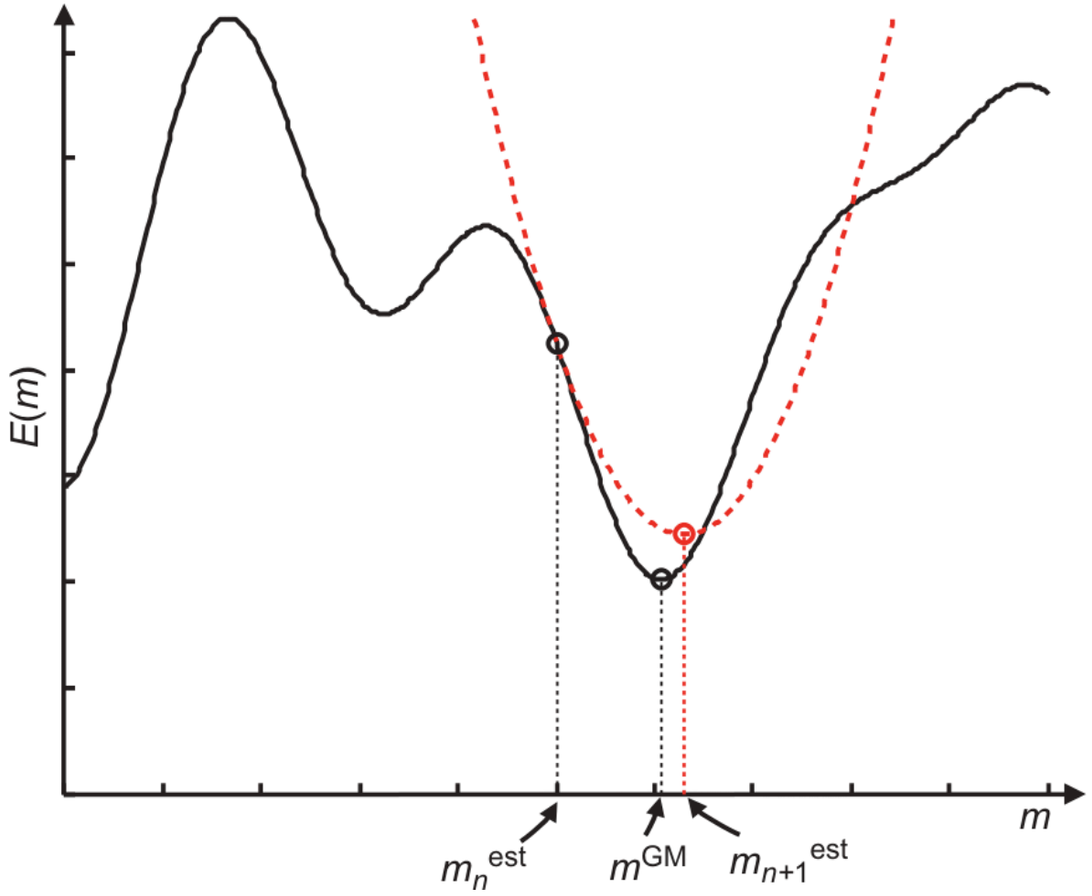


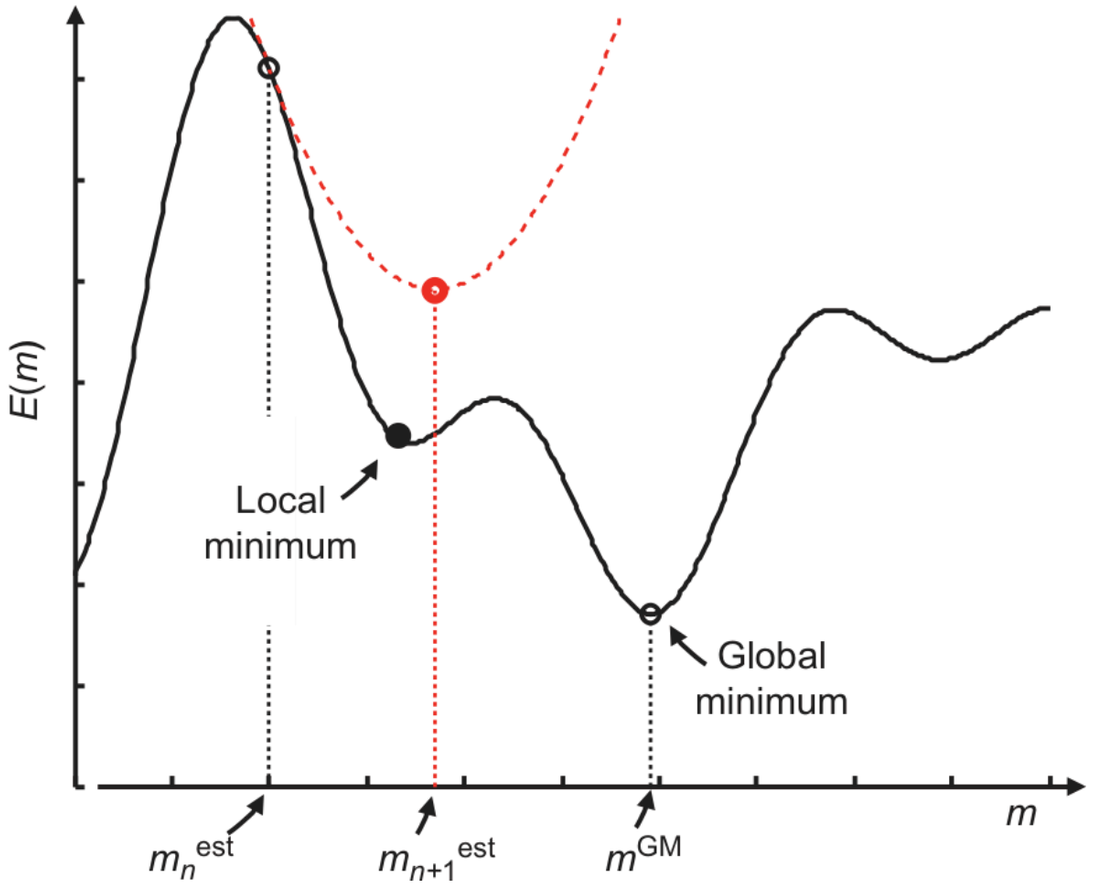
Monte Carlo method



Simulated Annealing

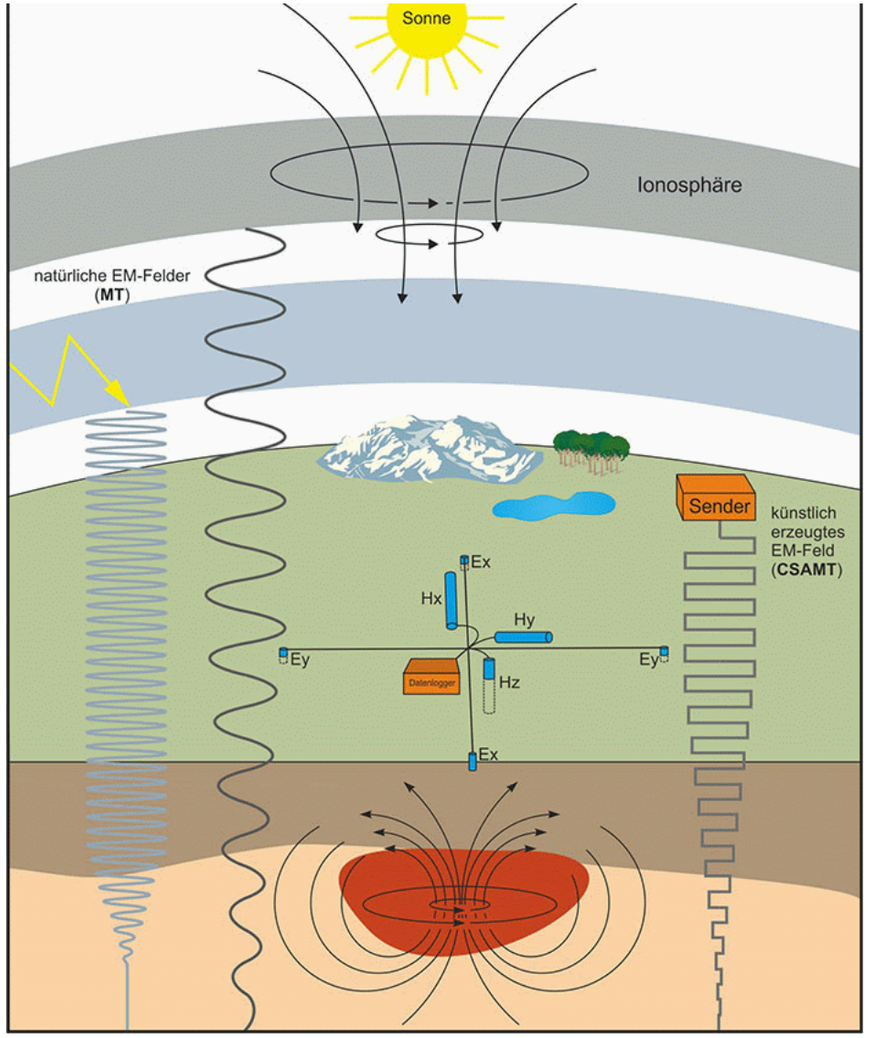
Newtons method (Menke, 2012)





linearize with value, slope and curvature of Φ_d

Problem: magnetotellurics



- inductive electromagnetics with f=0.001-100,Hz (T=0.01-1000s)
- source in ionosphere (natural source) or on ground (controlled source)
- measure magnetic and electric fields
- analyse (complex) ratio in frequency domain
- depth sounding (ρ_a & ϕ) with T

Magnetotelluric depth sounding

In magnetotellurics, we measure electric and magnetic fields over a wide frequency ($\omega = 2\pi/T$) or priod (T) range to make a depth sounding. From the impedance $Z = E/H$ (e.g. E_x and H_y) we compute the apparent resistivity

$$\rho^a(\omega) = \frac{1}{\omega\mu}|Z|^2$$

and phase

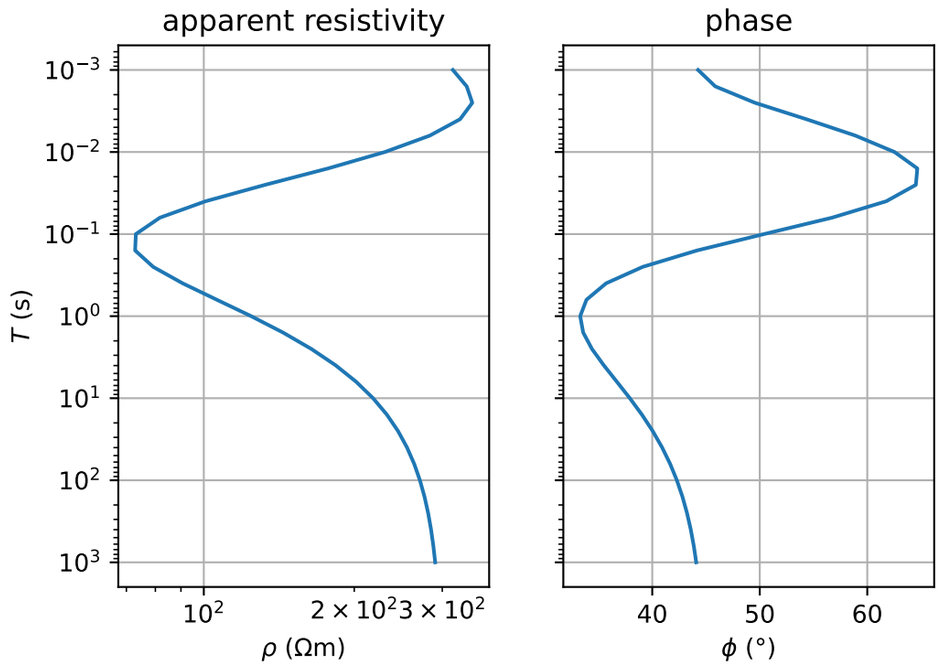
$$\phi(\omega) = \tan^{-1}(Z''/Z')$$

Implementation in Julia

```
function mt1dfwd(period, r, d)
    # MT forward computation with Wait's algorithm
    # = mt1dfwd(periods, resistivities,thicknesses)
    my0 = 4*pi*1e-7;
    omega = 2 *pi ./ period;
    k1 = sqrt.(im * omega * my0 / r[1]);
    g = ones(Complex, length(omega))
    nlay=min(length(r),length(d)+1)
    for k = nlay-1:-1:1;
        k1 = sqrt.(im * omega * my0 / r[k])
        k2 = sqrt.(im * omega * my0 / r[k+1])
        th = tanh.(k1*d[k])
        g = (g.*k2 + k1.*th) ./ (k1 + g.*k2.*th);
    end
    z = im * omega ./ (k1 .* g)
    rhoa = my0 ./ omega .* abs.(z).^2
    phi = angle.(z)
    return rhoa, phi
end;
```

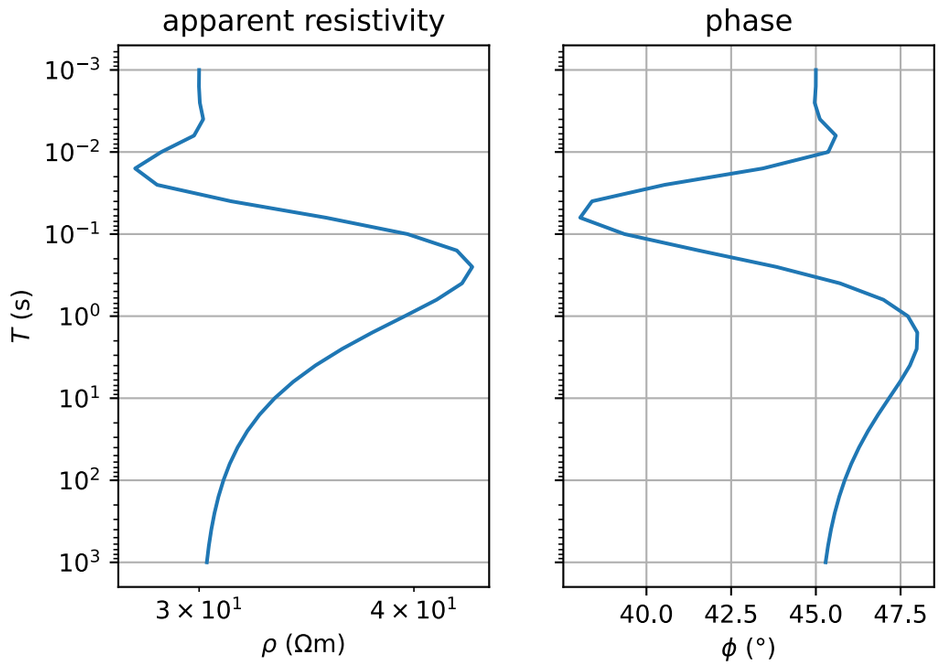
Implementation in Python

```
1  T = np.logspace(-3, 3, 31)
2  nlay = 3
3  fopBlock = pg.core.MT1dModelling(T, nlay
4  data = fopBlock(modelSynth)
5  modelSynth = [500, 500, 30, 300, 30]
6  data = fopBlock(modelSynth)
7  ax = plotMTsounding(T, data);
```



Bad conductor

```
T = np.logspace(-3, 3, 31)
nlay = 3
fopBlock = pg.core.MT1dModelling(T, nlay
data = fopBlock(modelSynth)
modelSynth = [500, 500, 300, 30, 300]
data = fopBlock(modelSynth)
ax = plotMTsounding(T, data);
```



1D (smooth) resistivity inversion

```
thk = np.logspace(1.5, 2.5, 15)
f = pg.core.MT1dRhoModelling(
    pg.Vector(T), pg.Vector(thk))
rho = np.ones(len(thk)+1) * np.median(rh
```

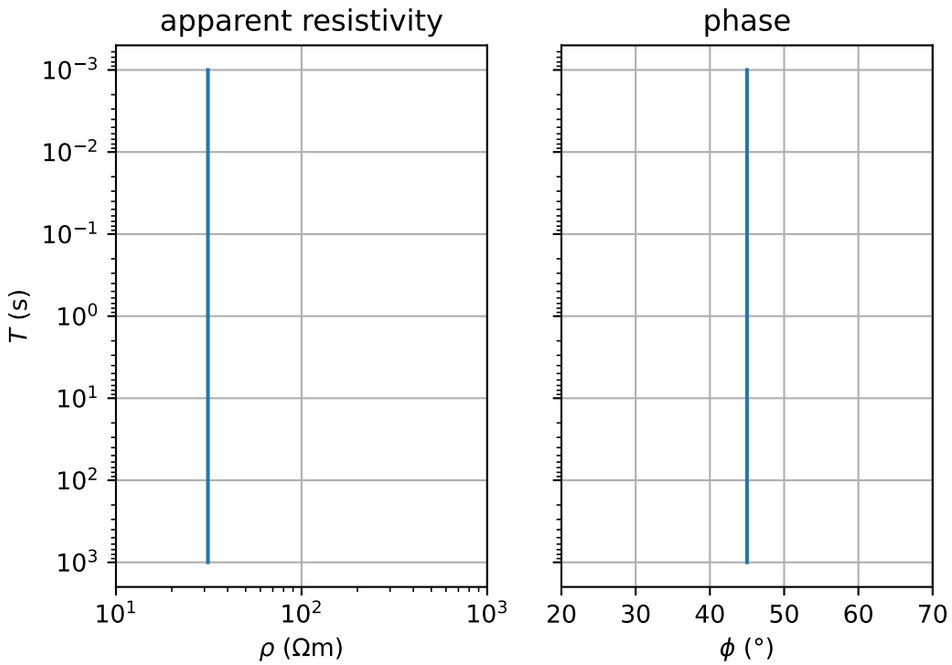
Let’s start

Forward operator pure resistivity

```
thk = np.logspace(1.5, 2.5, 15)
f = pg.core.MT1dRhoModelling(
    pg.Vector(T), pg.Vector(thk))
```

Starting model: halfspace

```
rho = np.ones(len(thk)+1) * np.median(rhoa)
response = f(rho)
ax = plotMTsounding(T, response);
```



1D (smooth) resistivity inversion

Gauss-Newton minimization

1. Choose starting model $\mathbf{m}^0$ and set n=0
2. Compute model response $\mathbf{f}(\mathbf{m}^n)$
3. Compute sensitivity matrix $\mathbf{S}^n$
4. Solve linearized subproblem $\mathbf{S}^n \Delta\mathbf{m}^n = \Delta\mathbf{d} = (\mathbf{d} - \mathbf{f}(\mathbf{m}^n))$
5. Optimize line search parameter τ^n
6. Update model by $\mathbf{m}^{n+1} = \mathbf{m}^n + \tau^n \Delta\mathbf{m}^n$
7. If convergence quit, otherwise $n \leftarrow n + 1$ & proceed with 2.

Computation of the sensitivity matrix

- analytically (derivation of the forward operator)
- transforming the PDE and its numerical solution
- perturbation method (brute force)

$$\frac{\partial f_i(\mathbf{m})}{\partial m_j} \approx \frac{f_i(\mathbf{m} + \delta_j \Delta m) - f_i(\mathbf{m})}{\Delta m}$$

with the Dirac vector $\delta_j = [0, \ldots, 0, 1, 0, \ldots, 0]^T$

$\Rightarrow$ one full forward computation for every model parameter

Combined model and data transformation

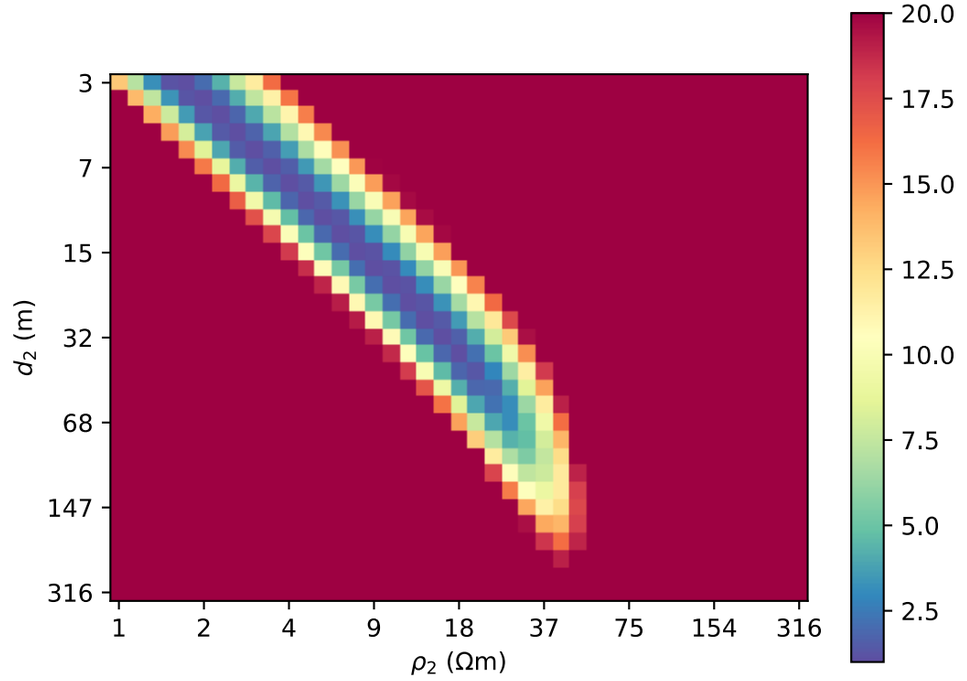
Sensitivity $S_{ij} = \frac{\partial \rho_i^a}{\partial \rho_j}$

Data $d_i = \log \rho_i^a$, model parameter $m_i = \log \rho_j$

$\Rightarrow$ Jacobian matrix $\frac{\partial \hat{f}}{\partial \hat{m}} = \frac{\partial f}{\partial m} \cdot \frac{\partial \hat{f}}{\partial f} \Big/ \frac{\partial \hat{m}}{\partial m}$

$$J_{ij} = \frac{\partial \log \rho_i^a}{\partial \log \rho_j} = \frac{\partial \rho_i^a}{\partial \rho_j} \cdot \frac{\rho_j}{\rho_i^a}$$

Model transformation



Objective function for VES

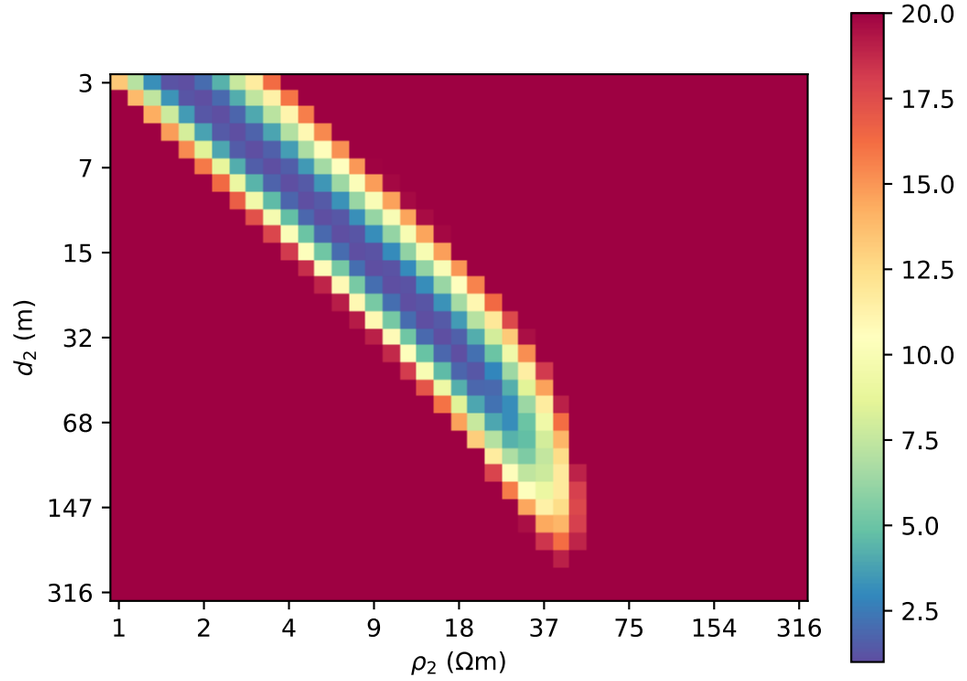
To keep the parameters positive, we often invert for the logarithms.

If we invert for $\hat{m}$ instead of m, we use the chain rule

$$\frac{\partial f}{\partial \hat{m}} = \frac{\partial f}{\partial m} \cdot \frac{\partial m}{\partial \hat{m}} = \frac{\partial f}{\partial m} \Big/ \frac{\partial \hat{m}}{\partial m}$$

E.g. $\partial \log \rho / \partial \rho = 1/\rho$

Data transformation



Objective function for VES

Often, measured data show a wide range so that we use the logarithm

If we invert $\hat{d}$ instead of d, we use the chain rule

$$\frac{\partial \hat{f}}{\partial m} = \frac{\partial f}{\partial m} \cdot \frac{\partial \hat{f}}{\partial f}$$

E.g. $\partial \log \rho^a / \partial \rho^a = 1/\rho_a$