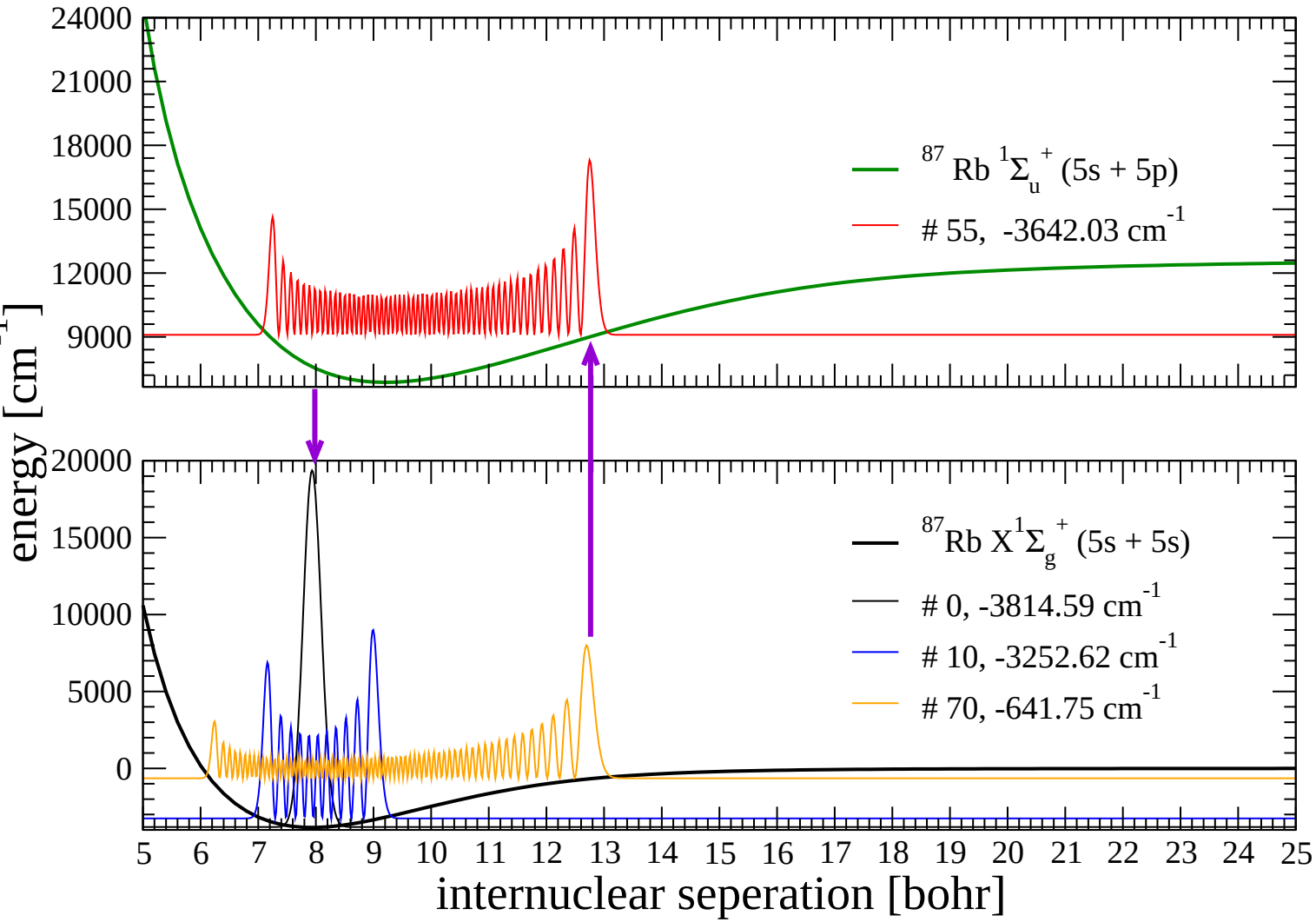
# Comparison of GRAPE/LBFGS and Krotov in a High-Dimensional Hilbert Space

Michael Goerz
FU Berlin
C. Koch Group

OCT Comparison Workshop
Cambridge, UK
November 26, 2010

# Coherent Transfer between Vibrational States in Rb2



# Outline

# Grape and LBFGS

# Grape

## Acronym

GRAPE: **G**radient **A**scent **P**ulse **E**ngineering

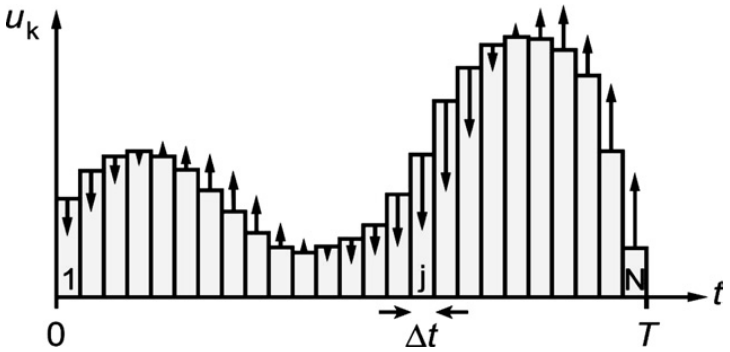
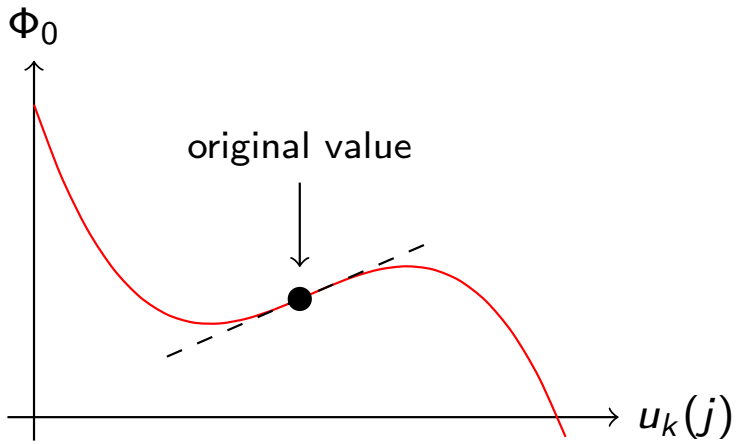



Fig. 1. Schematic representation of a control amplitude $u_k(t)$, consisting of $N$ steps of duration $\Delta t = T/N$. During each step $j$, the control amplitude $u_k(j)$ is constant. The vertical arrows represent gradients $\delta\Phi_0/\delta u_k(j)$, indicating how each amplitude $u_k(j)$ should be modified in the next iteration to improve the performance function $\Phi_0$.




at time index $j$: go in direction of gradient

## Pulse Update

$$u_k(j) \longrightarrow u_k(j) - \epsilon \frac{\partial \Phi_0}{\partial u_k(j)}$$

# Second Derivative: Newton's Method

## Newton's method (one-dimensional case)

$$f(x_0 + \Delta x) = f(x_0) + f'(x_0) \cdot \Delta x + \frac{1}{2} f''(x_0) \cdot (\Delta x)^2$$

$$\frac{\mathrm{d}f(x_0 + \Delta x)}{\mathrm{d}(\Delta x)} = 0 \qquad \Rightarrow \qquad x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

# Second Derivative: Newton's Method

## Newton's method (one-dimensional case)

$$f(x_0 + \Delta x) = f(x_0) + f'(x_0) \cdot \Delta x + \frac{1}{2} f''(x_0) \cdot (\Delta x)^2$$

$$\frac{\mathrm{d}f(x_0 + \Delta x)}{\mathrm{d}(\Delta x)} = 0 \qquad \Rightarrow \qquad x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

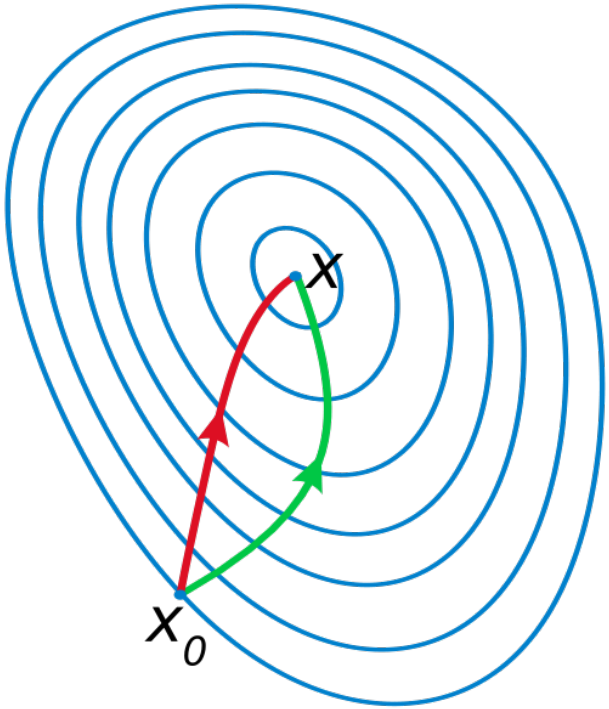
## Newton's method (multi-dimensional case)

Sequence

$$\vec{x}_{n+1} = \vec{x}_n - H_f^{-1}(\vec{x}_n) \cdot \vec{\nabla} f(\vec{x}_0)$$

converges towards extremum.

# Gradient Descent and Newton's Method




Optimization towards an extremal point by gradient descent (green) and Newton's method (red)

# Quasi-Newton

## Newton's method (multi-dimensional case)

Sequence

$$\vec{x}_{n+1} = \vec{x}_n - H_f^{-1}(\vec{x}_n) \cdot \vec{\nabla} f(\vec{x}_0)$$

converges towards extremum.

Find matrix $B$ as approximation to $H^{-1}$ so that $B$ fulfills the secant equation.

## Secant Equation

$$\frac{f'(x_0 + \Delta x) - f'(x_0)}{(x_0 + \Delta x) - x_0} = B$$

Underdetermined in higher dimensions! LBFGS is one option to construct $B$.

# Quasi-Newton Algorithms

## Quasi-Newton method (general)

Given a $\vec{x}_0 \in \mathbb{R}^N$ chosen sufficiently close to a local extremum $\vec{x}_E$ of $f$ and an initial guess for the Hessian $B_0$ (for example $B_0 = I$) repeat the following steps to obtain $\vec{x}_E$:

1. Calculate the step $\Delta\vec{x}_k$ using the current approximated Hessian $B_k$ by: $\Delta\vec{x}_k = -B_k^{-1} \cdot \vec{\nabla} f(\vec{x}_k)$.
2. Calculate the new $\vec{x}_{k+1}$: $\vec{x}_{k+1} = \vec{x}_k + \Delta\vec{x}_k$.
3. Use the gradient at the new point $\vec{\nabla} f(\vec{x}_{k+1})$ and the difference in gradients between new and old point: $\vec{y}_k = \vec{\nabla} f(\vec{x}_{k+1}) - \vec{\nabla} f(\vec{x}_k)$ to find a new approximation for the Hessian $B_{k+1}$.

## Linesearch

Introduce parameter $\alpha_k \in [0, 1]$ to modify step size:

$$\Delta\vec{x}_k = -\alpha_k \cdot B_k^{-1} \cdot \vec{\nabla} f(\vec{x}_k)$$

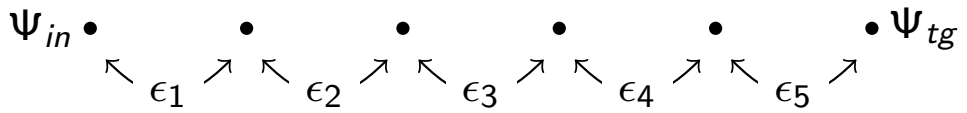
# LBFGS

- Quasi-Newton algorithms are approximate solutions to an extremization problem using information from the second order Taylor expansion of the function
- BFGS is a quasi-Newton algorithm using a rank-two update formula involving only gradients to the Hessian needed to determine the update direction; for convex functions it is globally and monotonic convergent if one enhances it by line search fulfilling the Wolfe conditions
- L-BFGS uses only information from the gradients and point vectors of previous steps to solve the memory problem in storing the BFGS approximated Hessian - under certain additional assumptions the convergence behavior stays intact

An LBFGS implementation is available as a free Fortran 77 library:

Zhu et al. *Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization*. ACM Trans. Math. Softw. **23**, 550 (1997)
`http://portal.acm.org/citation.cfm?id=279236`

# Calculating the Gradient




$$J = -F(\Psi(T)) + \int_{t=0}^{T} g_a(\epsilon(t))\,\mathrm{d}t + \int_{t=0}^{T} g_b(\Psi,\epsilon))\,\mathrm{d}t$$

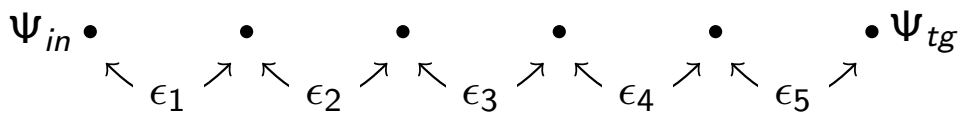
$$F = \frac{1}{N}\Re\,\mathrm{tr}\left\{\hat{O}^\dagger \hat{U}(T,0)\right\} = \frac{1}{N}\Re\sum_{k=1}^{N}\left\langle \Psi_{tg}\left|\hat{U}(T,0)\right|\Psi_{in}\right\rangle$$

$$g_a = \frac{\alpha}{S(t)}\left(\epsilon - \epsilon^{\mathrm{old}}\right)^2$$

**Gradient for every pulse value**

$$G_i = \frac{\partial J}{\partial \epsilon_i} = -\frac{\partial F}{\partial \epsilon_i} + \frac{\partial}{\partial \epsilon_i}\sum_i \frac{\alpha}{S_i}\left(\epsilon_i - \epsilon_i^{\mathrm{old}}\right)^2 \Delta t$$

# Calculating the Gradient




$$\frac{\partial}{\partial \epsilon_2} F = \Re \left\langle \Psi_{tg} \left| \hat{U}_5 \hat{U}_4 \hat{U}_3 \frac{\partial \hat{U}_2}{\partial \epsilon_2} \hat{U}_1 \right| \Psi_{in} \right\rangle ; \qquad \hat{U}_i = e^{-i\hat{H}(\epsilon_i)\Delta t}$$

$$= \Re \left\langle \Psi_{bw} \left| \frac{\partial \hat{U}_2}{\partial \epsilon_2} \right| \Psi_{fw} \right\rangle$$

$$\frac{\partial \hat{U}_i}{\partial \epsilon_i} = \frac{\partial}{\partial \epsilon_i} e^{-i\hat{H}(\epsilon_i)\Delta t}$$

$$= \sum_{n=1}^{\infty} \frac{(-i\Delta t)^n}{n!} \sum_{k=0}^{n-1} \hat{H}^k \frac{\partial H(\epsilon_i)}{\partial \epsilon_i} \hat{H}^{n-k-1}$$

# The Basic Idea of Krotov: Convergence by Construction

Ingredients:

- final-time target $J_T[\varphi_T, \varphi_T^*]$
- time-dep. targets / costs $g_a[\epsilon] + g_b[\varphi(t), \varphi^*(t)]$
- equations of motion $i\hbar \frac{\partial}{\partial t}|\varphi(t)\rangle = \hat{\mathbf{H}}(t)|\varphi(t)\rangle \quad |\varphi(t_0)\rangle = |\varphi_0\rangle$

Construction of **auxiliary functional** $L$

$$L[\varphi, \varphi^*, \epsilon, \Phi] = J[\varphi, \varphi^*, \epsilon]$$

choose **arbitrary scalar potential** $\Phi[\varphi, \varphi^*, t]$ such that

$$L[\varphi^i, \varphi^{*,i}, \epsilon^i, \Phi] \geq L[\varphi^{i+1}, \varphi^{*,i+1}, \epsilon^{i+1}, \Phi]$$

$\longrightarrow$ *building in monotonic convergence*

# Auxiliary functional $L$

$$L[\varphi, \varphi^*, \epsilon, \Phi] = G[\varphi(T), \varphi^*(T)] - \Phi[\varphi(0), \varphi^*(0), 0] - \int_0^T R[\varphi(t), \varphi^*(t), \epsilon(t), t]dt$$

**final-time contribution:**

$$G[\varphi(T), \varphi^*(T)] = J_T[\varphi(T), \varphi^*(T)] + \Phi[\varphi(T), \varphi^*(T), T]$$

**intermediate-time contribution:**

$$R[\varphi(t), \varphi^*(t), \epsilon(t), t] = -\Big(g_a[\epsilon(t)] + g_b[\varphi(t), \varphi^*(t)]\Big) + \frac{\partial \Phi}{\partial t} + \sum_{k=1}^{N} \Big[\nabla_{\varphi_k} \Phi \cdot f_k[\varphi, \varphi^*, \epsilon, t] + \nabla_{\varphi_k^*} \Phi \cdot f_k^*[\varphi, \varphi^*, \epsilon, t]\Big]$$

# Central Idea of Krotov's Method

We want a minimum of $L$, i.e. minimum of $G$ & maximum of $R$
**but** $L$ is changed by both changes in $\vec{\varphi}$ and changes in $\epsilon$

## Krotov's Solution

(i) choose $\Phi$ at the extremum, $\vec{\varphi}^i$, such that it is the **worst** possible choice with respect to any change in the states $\curvearrowright$ maximize $L$ when going from $\vec{\varphi}^i$ to $\vec{\varphi}^{i+1}$ for fixed $\epsilon^i$

(ii) then any change in the field from $\epsilon^i$ to $\epsilon^{i+1}$ will lead to a minimization of $L$

$$\epsilon^{(i+1)}(t) = \arg\max_{\epsilon(t)} R\left(\vec{\varphi}(t)^{(i+1)}, \epsilon(t), t\right) \quad \text{or}$$

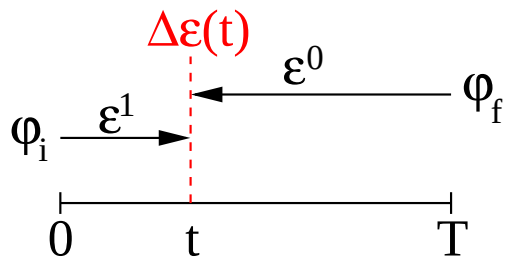
$$\frac{\partial R}{\partial \epsilon}\left(\vec{\varphi}^{(i+1)}, \epsilon^{(i+1)}, t\right) = 0 \quad , \quad \frac{\partial^2 R}{\partial \epsilon^2}\left(\vec{\varphi}^{(i+1)}, \epsilon^{(i+1)}, t\right) < 0$$

# Pulse Update by Krotov

## Krotov Update Formula

$$\Delta\epsilon(t) = \frac{S(t)}{\alpha} \Im \left[ \sum_{k=1}^{N} a_k \left\langle \Psi_{in,k} \left| \hat{O}^\dagger \hat{U}^\dagger(T \to t, \epsilon^{(i)}) \hat{\mu} \hat{U}(0 \to t, \epsilon^{(i+1)}) \right| \Psi_{in,k} \right\rangle \right]$$

*interference between past and future events*



# Second Order Krotov

## Second Order Krotov Update Formula

$$\Delta\epsilon(t) = \frac{S(t)}{\alpha} \Im \left[ \sum_{k=1}^{N} \left\langle \chi_k^{(i)}(t) \left| \frac{\partial \hat{H}}{\partial \epsilon} \right| \phi_k^{(i+1)}(t) \right\rangle \right] \quad \leftarrow \textit{first order}$$

$$\textit{second order} \rightarrow \quad + \frac{\sigma(t)}{2} \sum_{k=1}^{N} \left\langle \Delta\phi_k^{(i+1)}(t) \left| \frac{\partial \hat{H}}{\partial \epsilon} \right| \phi_k^{(i+1)}(t) \right\rangle \Bigg]$$

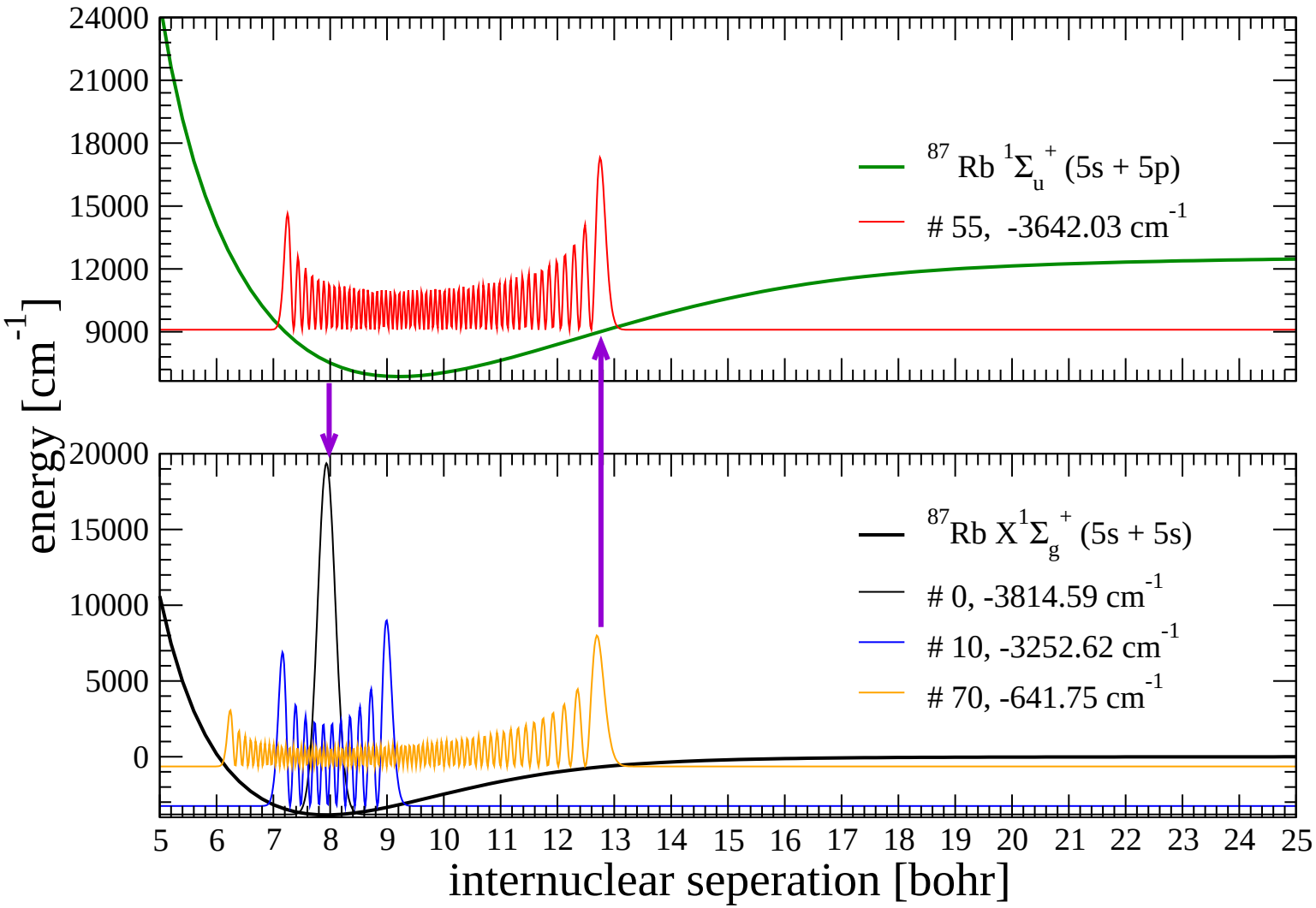
## Second Order $\sigma(t)$

$$\sigma(t) = \begin{cases} e^{\bar{B}(T-t)} \left( \frac{\bar{C}}{\bar{B}} - \bar{A} \right) \frac{\bar{C}}{\bar{B}} & \text{for } \bar{B} \neq 0 \\ \bar{C}(T-t) - \bar{A} & \text{for } \bar{B} = 0 \end{cases}$$

Daniel Reich, Mamadou Ndong and Christiane P. Koch
Monotonically convergent optimization in quantum control using Krotov's method.
**arXiv:1008.5126**

# Coherent Transfer between Vibrational States in Rb2

# Coherent Transfer between Vibrational States in Rb2



# Questions for LBFGS

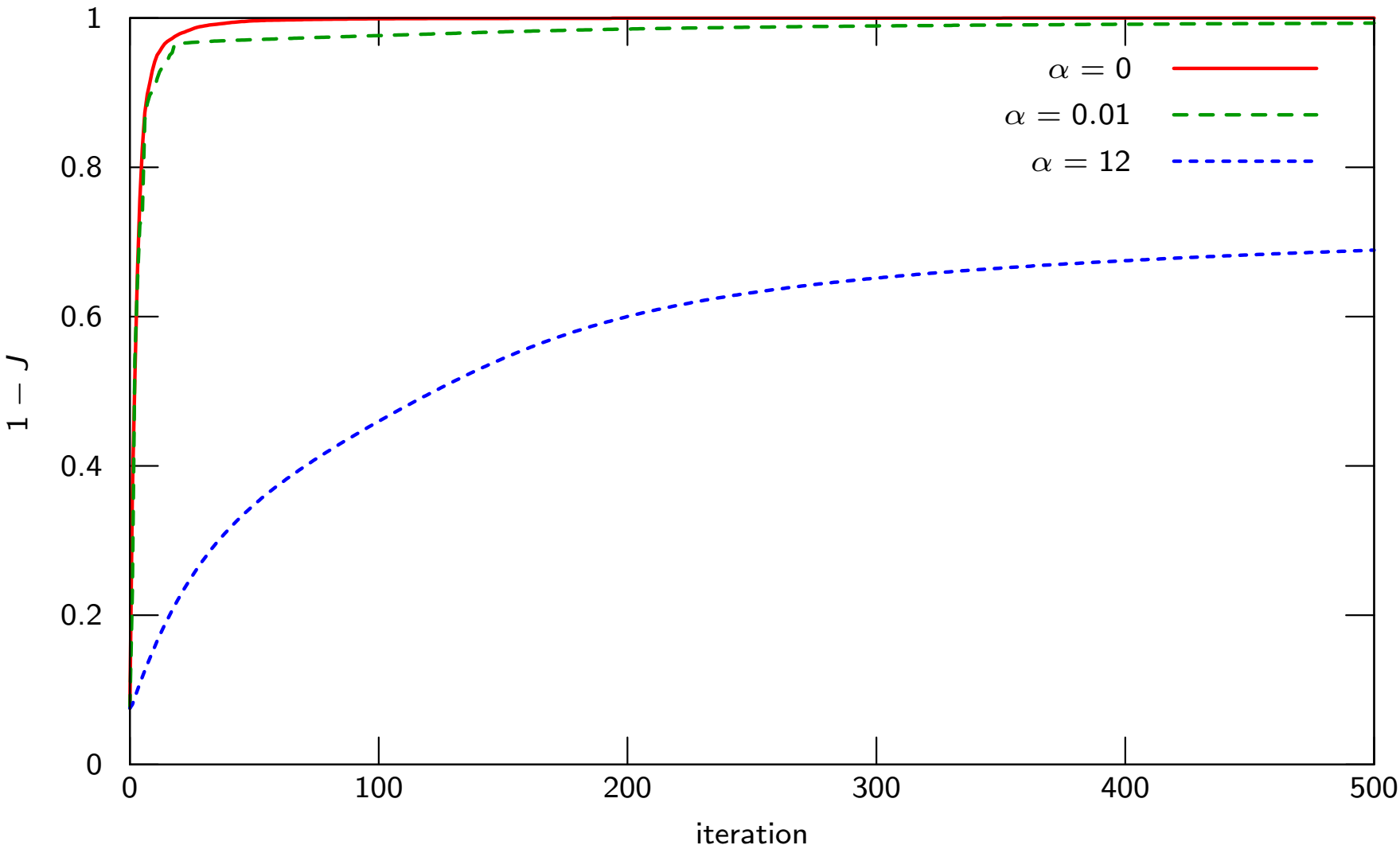
## Cost Functional

$$J = -\left\langle \Psi_{tg} \left| \hat{U}(T,0) \right| \Psi_{in} \right\rangle + \int_{t=0}^{T} g_a(\epsilon(t))\,\mathrm{d}t$$

- Can I include the running cost?
- Which order of the gradient do I need?
- How much does LBFGS improve on Grape?
- How does LBFGS compare to Krotov?

Simple Problem: $v = 10 \rightarrow v = 0$

# Running cost with LBFGS ($v = 10 \rightarrow v = 0$)



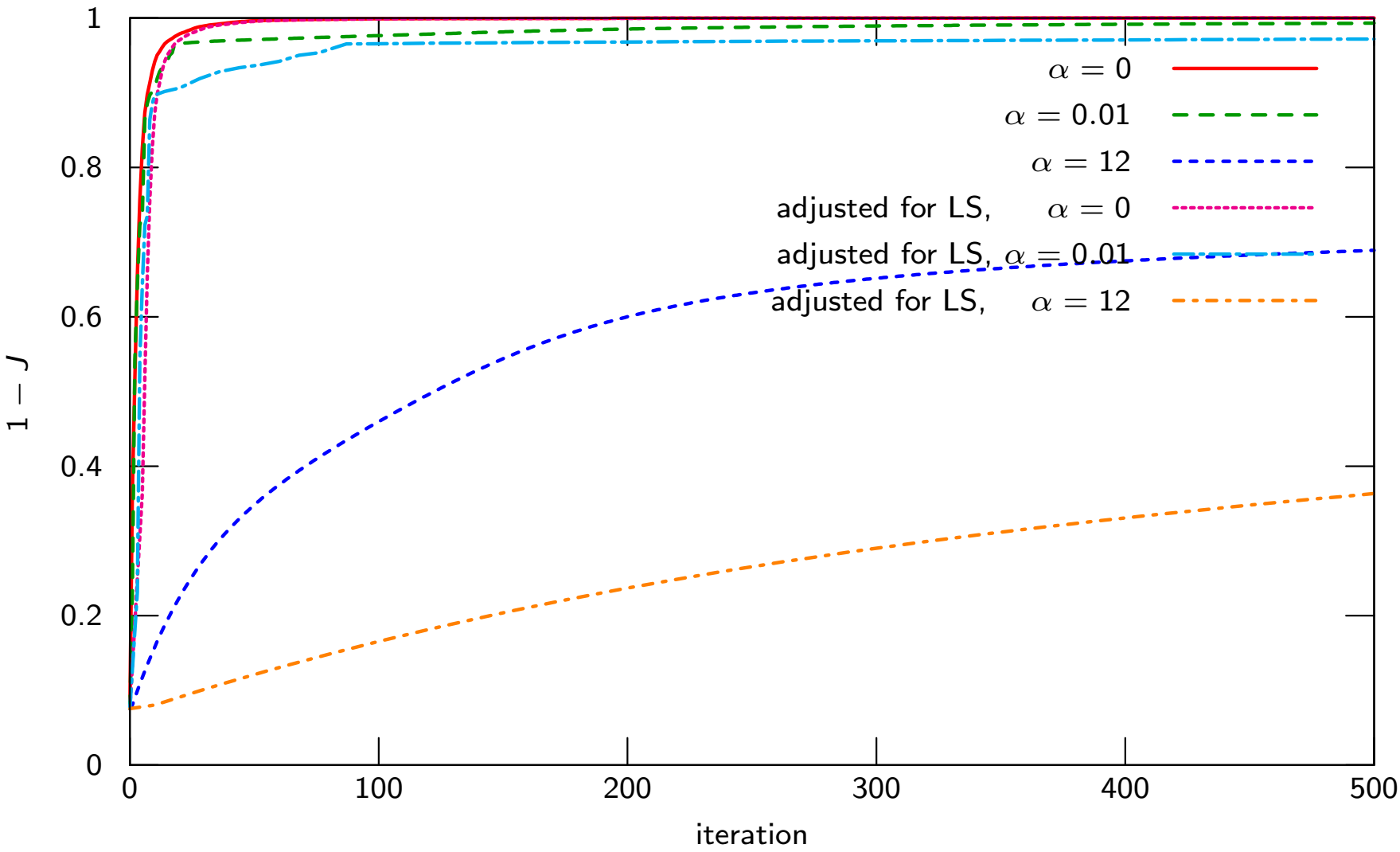
# Remember the Linesearch!

xterm

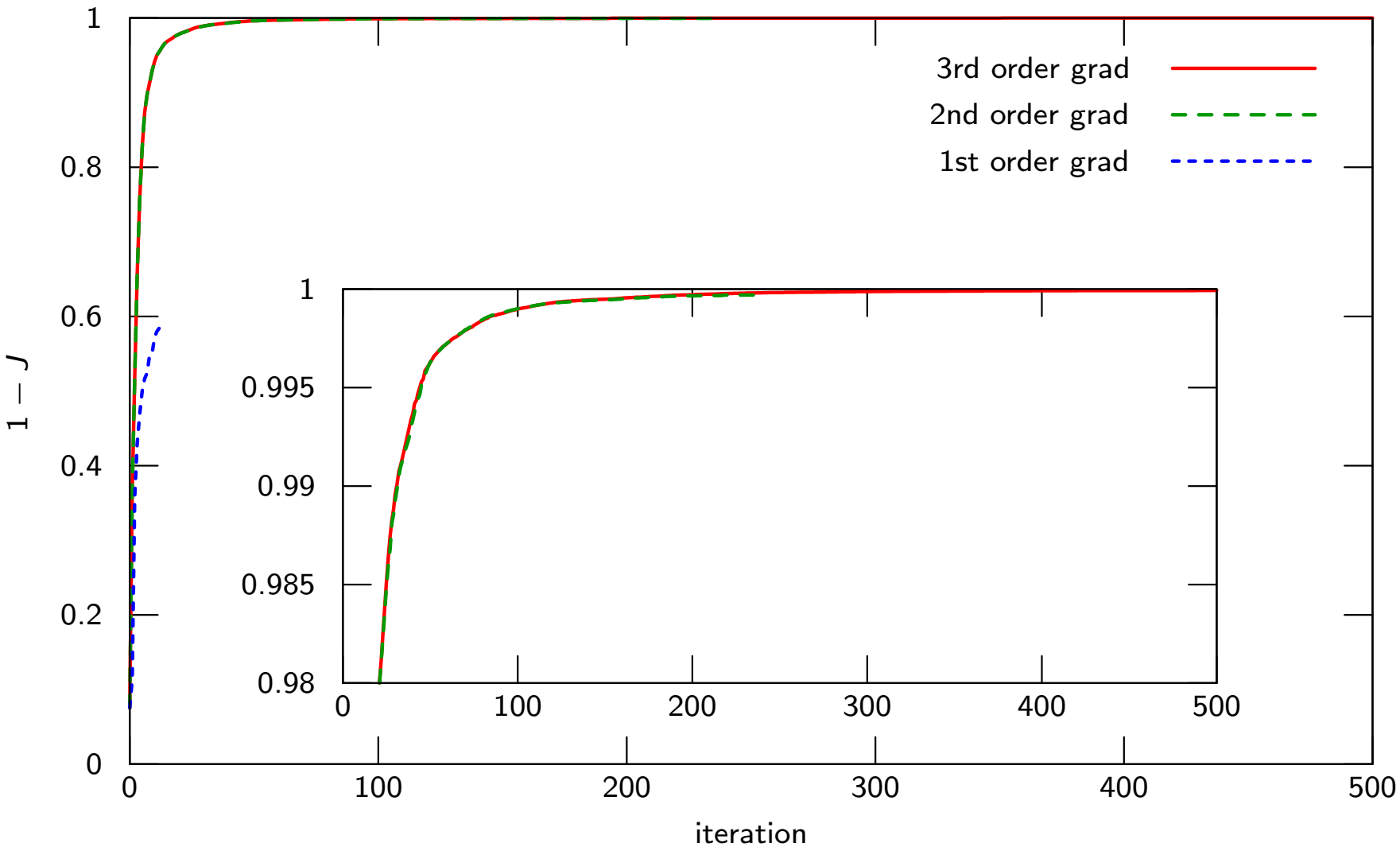
Default | Aramis

```
  *** Do the OCT ***
 Tue Nov 23 23:47:52 +0100 2010
 Initializing LBFGS
RUNNING THE L-BFGS-B CODE

           * * *

Machine precision = 2.220D-16
 N =        20672     M =           10
 This problem is unconstrained.
 Done with LBFGS Initialization

Iter. | Fidelity |  Infidelity |     g_a_int |     g_b_int |           J | LS |
    0 | 0.020042 | 9.79958E-01 | 0.00000E+00 | 0.00000E+00 | -2.00416E-02 |  0 |
    1 | 0.020088 | 9.79912E-01 | 2.30396E-05 | 0.00000E+00 | -2.00645E-02 | 10 |
    2 | 0.020133 | 9.79867E-01 | 2.25744E-05 | 0.00000E+00 | -2.01100E-02 |  9 |
    3 | 0.020178 | 9.79822E-01 | 2.25761E-05 | 0.00000E+00 | -2.01550E-02 |  9 |
    4 | 0.020223 | 9.79777E-01 | 2.25774E-05 | 0.00000E+00 | -2.02001E-02 |  9 |
    5 | 0.020268 | 9.79732E-01 | 2.25786E-05 | 0.00000E+00 | -2.02451E-02 |  9 |
    6 | 0.020313 | 9.79687E-01 | 2.25799E-05 | 0.00000E+00 | -2.02901E-02 |  9 |
    7 | 0.020358 | 9.79642E-01 | 2.25811E-05 | 0.00000E+00 | -2.03351E-02 |  9 |
    8 | 0.020403 | 9.79597E-01 | 2.25824E-05 | 0.00000E+00 | -2.03802E-02 |  9 |
    9 | 0.020448 | 9.79552E-01 | 2.25836E-05 | 0.00000E+00 | -2.04252E-02 |  9 |
   10 | 0.020493 | 9.79507E-01 | 2.25849E-05 | 0.00000E+00 | -2.04702E-02 |  9 |
goerz@n044:/local_scratch/132176.torque.cluster.physik.fu-berlin.de$
```

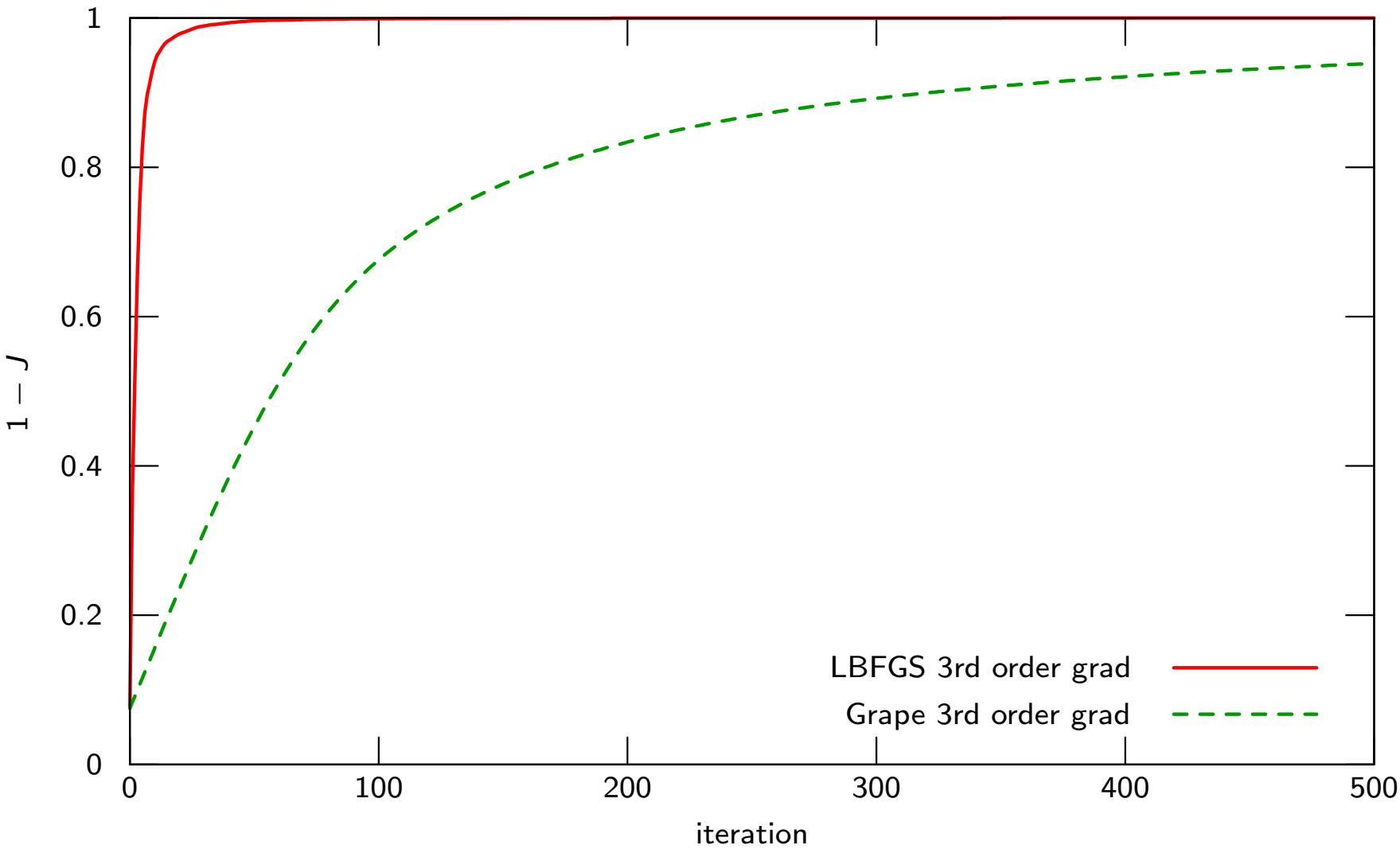
# Running cost with LBFGS ($v = 10 \rightarrow v = 0$)



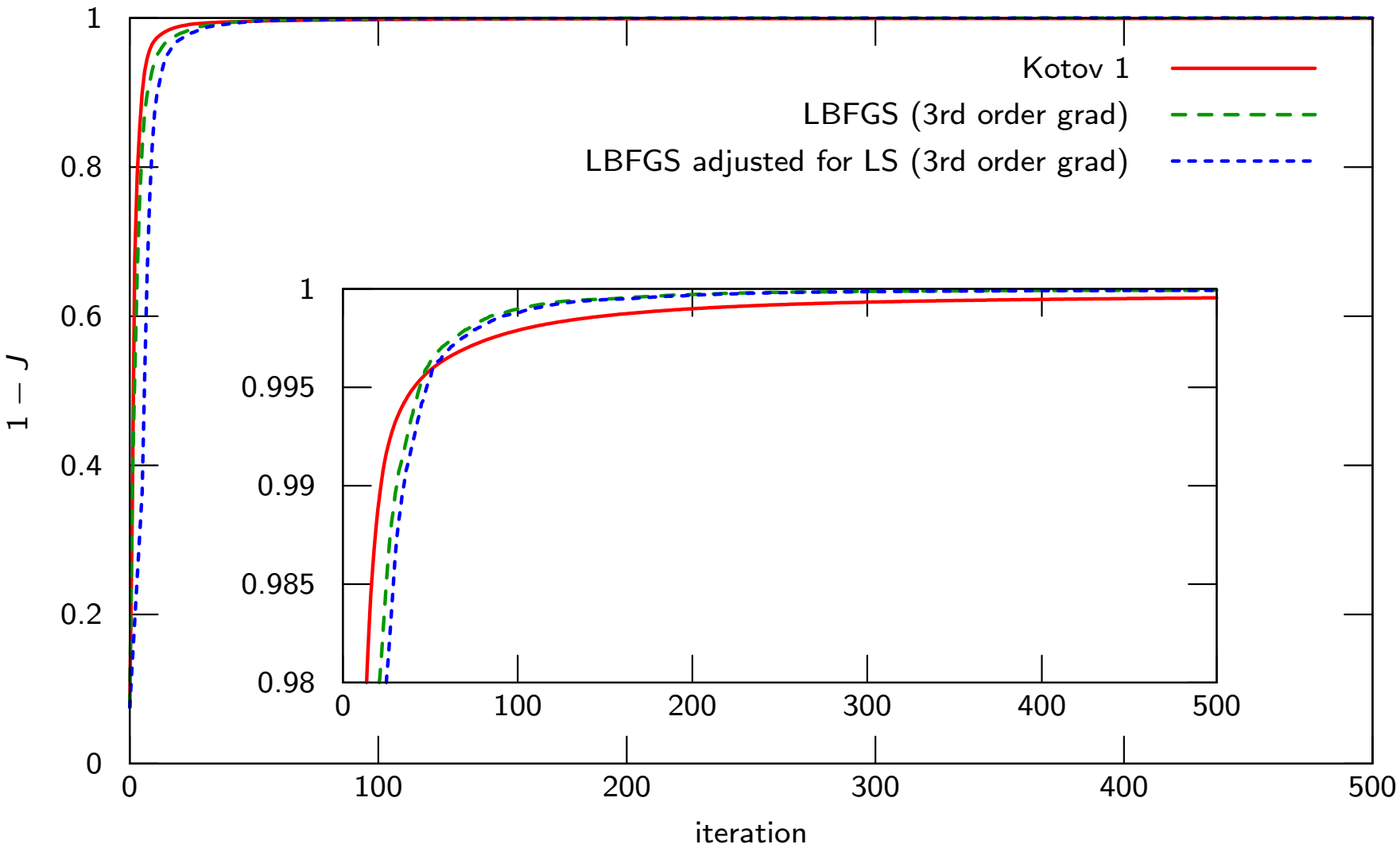
# Order of the Gradient in LBFGS ($v = 10 \rightarrow v = 0$)



# LBFGS vs Grape ($v = 10 \rightarrow v = 0$)



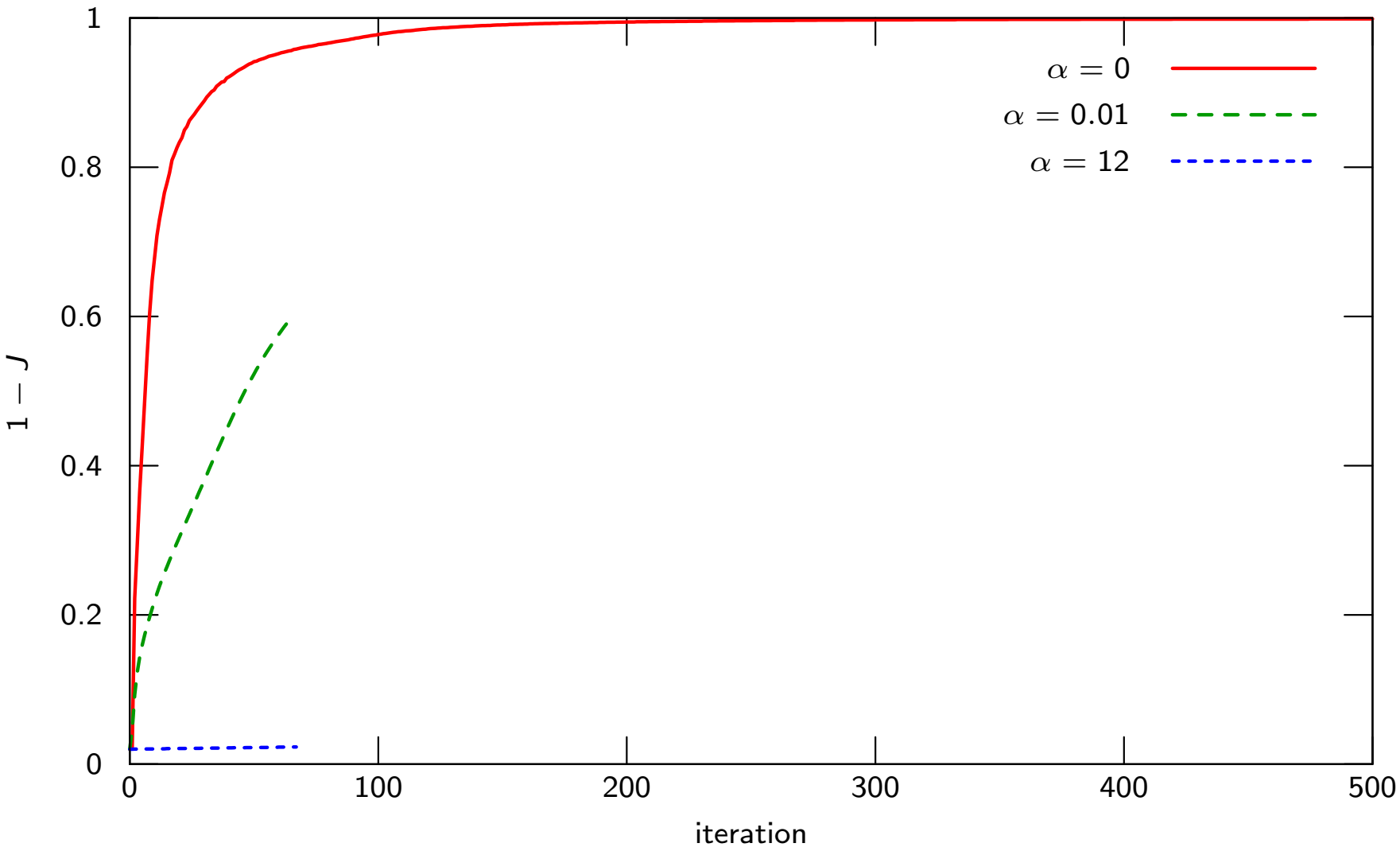
# Krotov vs LBFGS ($v = 10 \rightarrow v = 0$)



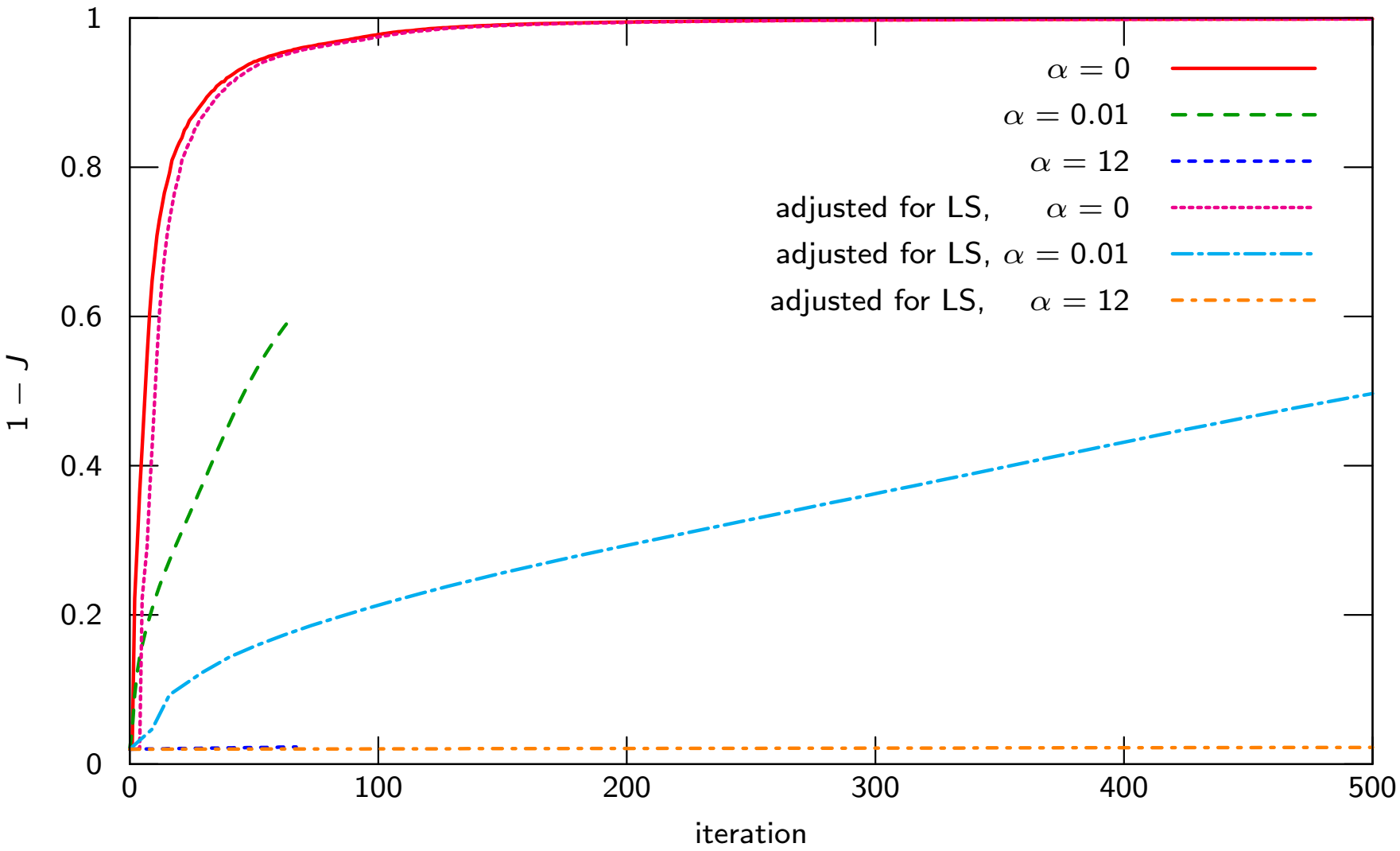
- Can I include the running cost? — No
- Which order of the gradient do I need? — At least second order
- How much does LBFGS improve on Grape? — A lot. (Forget about Grape)
- How does LBFGS compare to Krotov? — Not too shabby

# More Advanced Problem: $v = 70 \rightarrow v = 0$

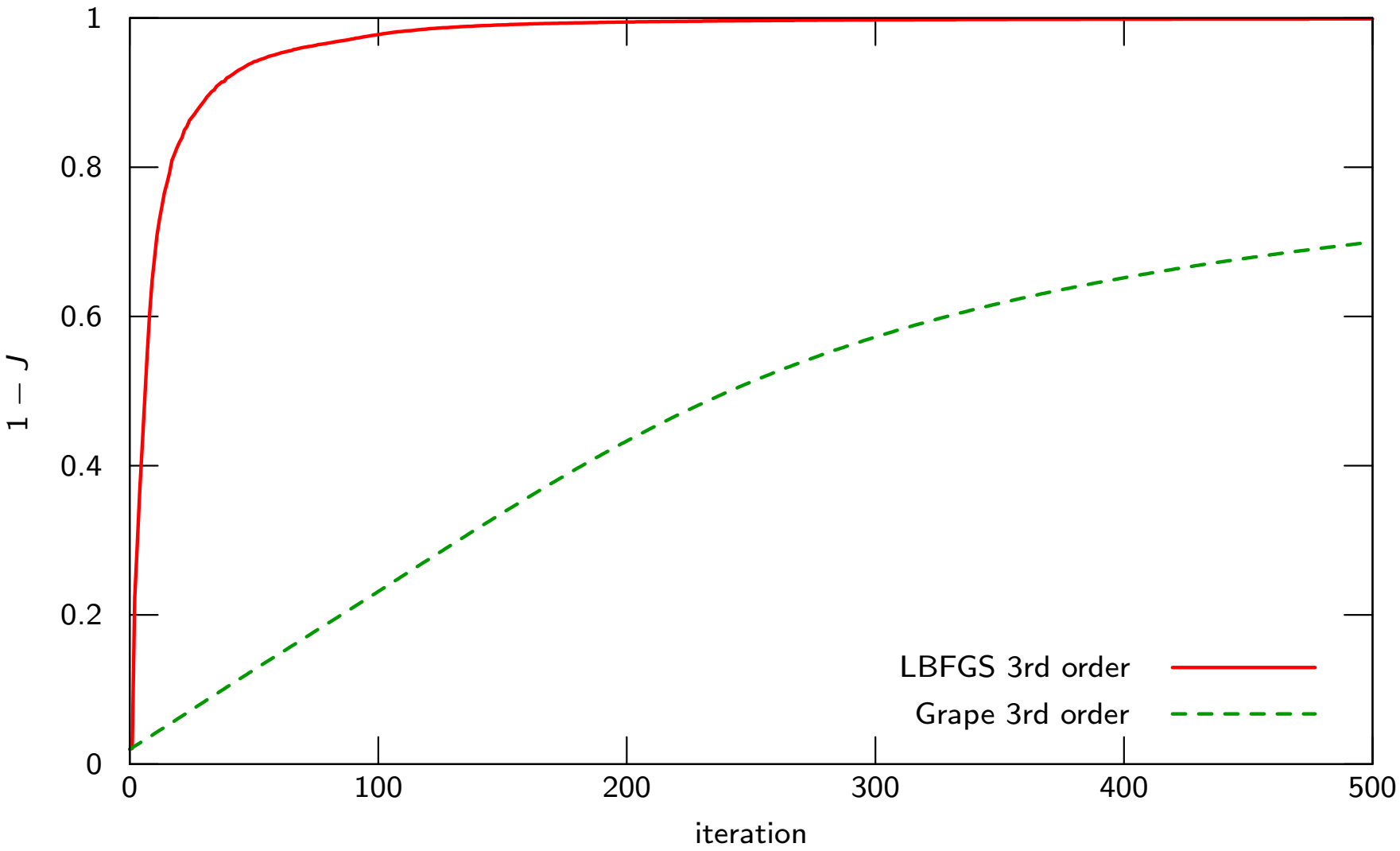
# Running cost with LBFGS ($v = 70 \rightarrow v = 0$)



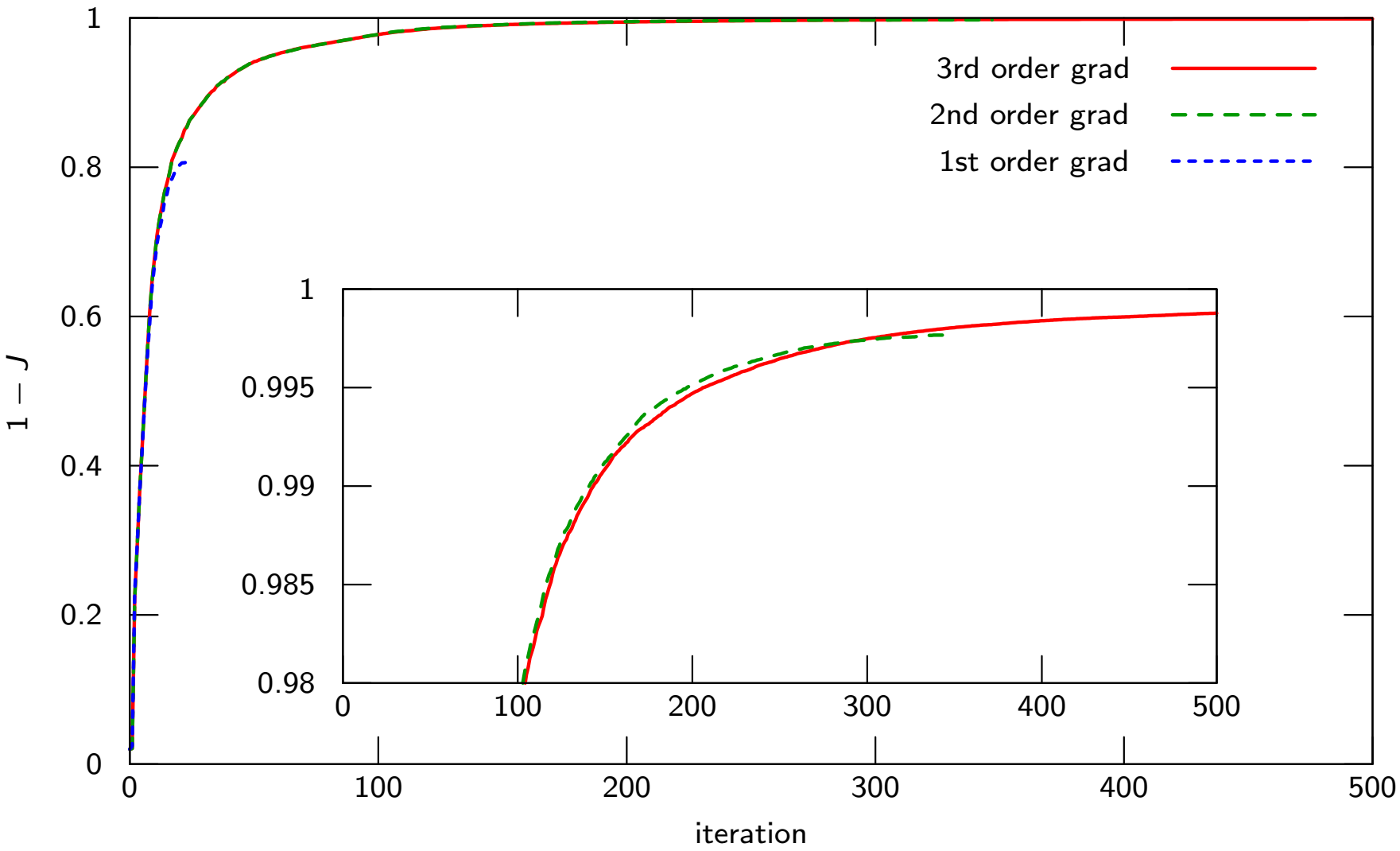
# Running cost with LBFGS ($v = 70 \rightarrow v = 0$)



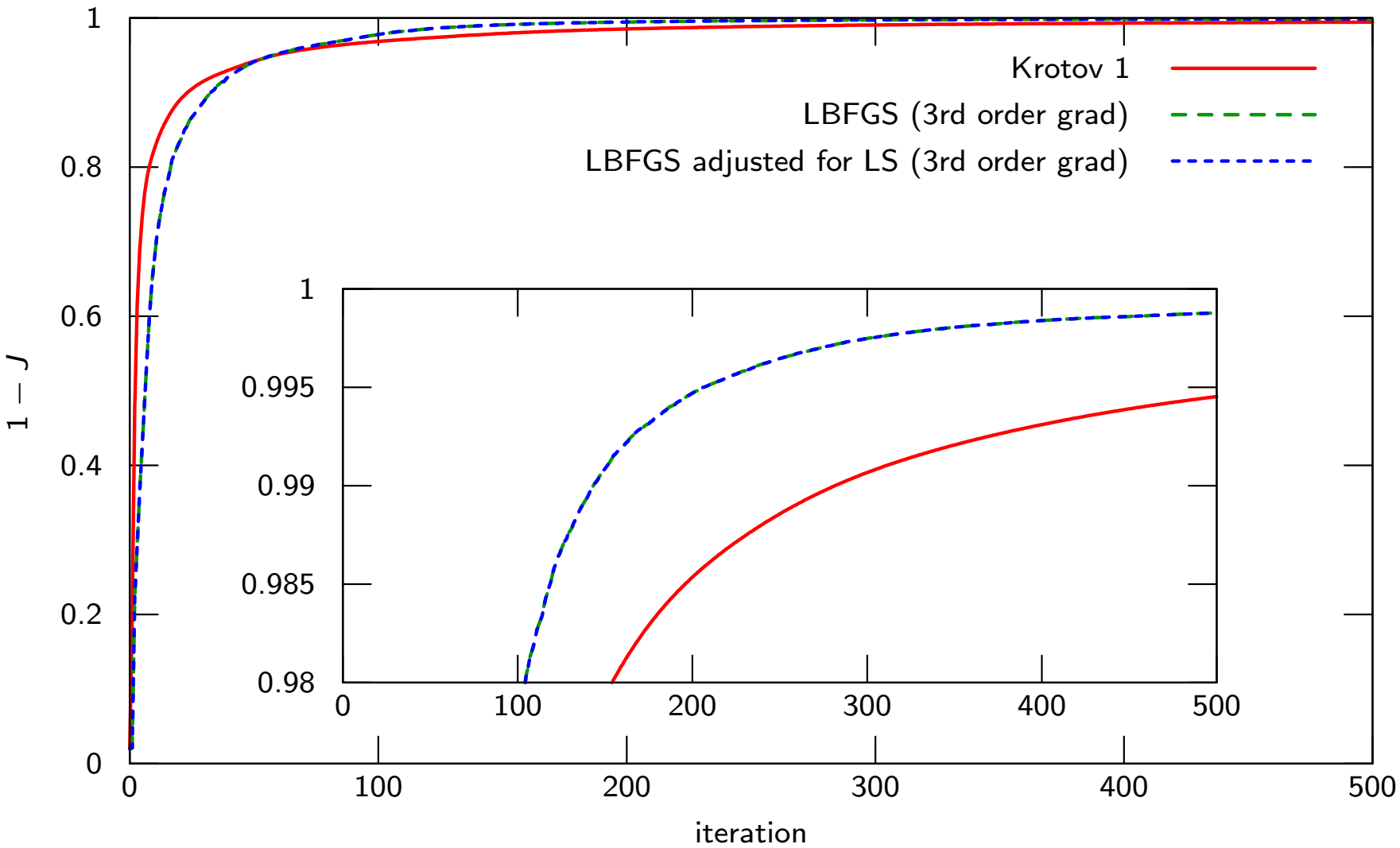
# LBFGS vs Grape ($v = 70 \rightarrow v = 0$)



# Order of the Gradient in LBFGS ($v = 70 \rightarrow v = 0$)



# Krotov vs LBFGS ($v = 70 \rightarrow v = 0$)



# Krotov Second Order

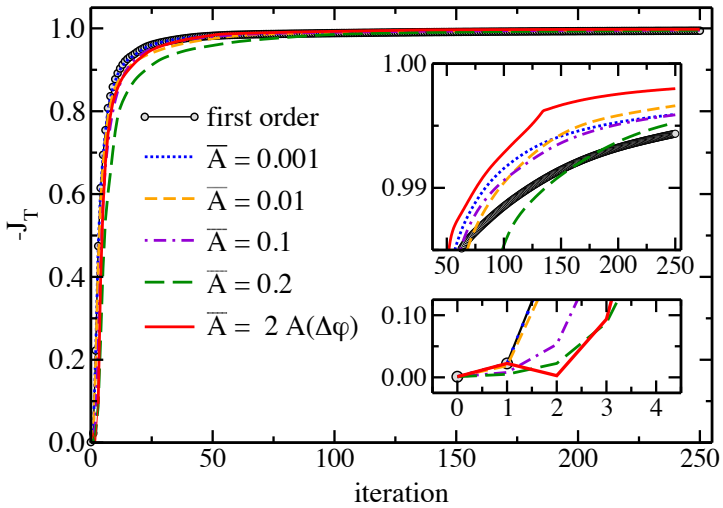



FIG. 2: (color online) Convergence of the first order and second order constructions of the optimization algorithm as measured by the final-time objective, $J_T$, for state-to-state transfer from vibrational level $v = 10$ to $v = 0$.

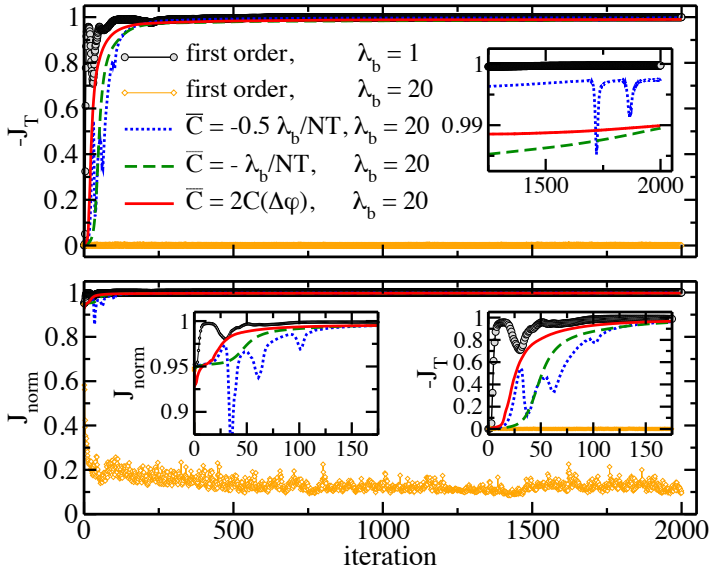



FIG. 5: (color online) Convergence of the first order and second order constructions for state-to-state transfer with state-dependent cost. The operator $\hat{\mathbf{D}}$ is taken to be the projector onto a forbidden subspace, i.e. the second order construction *is* required.

Daniel Reich, Mamadou Ndong and Christiane P. Koch
Monotonically convergent optimization in quantum control using Krotov's method.
**arXiv:1008.5126**

# Conclusions

- Use LBFGS if gradient can be calculated easily. Watch the number of linesearches if propagation is expensive.
- Make sure the gradient is exact enough (at least second order)
- Use Krotov second order when cost functional requires it, e.g. with state-dependent costs.
- **Combine Krotov with LBFGS?**

# Thank You!

**AG Koch — Moving from Berlin to Kassel!**

- Christiane Koch, already at Kassel
- Daniel Reich
- Mamadou Ndong, now at Laboratoire de Chimie Physique d'Orsay
- Ruzin Ağanoğlu
- Anton Haase