

THEORY OF TENSOR-TRAIN QUANTUM DYNAMICS AND OPTIMIZATION

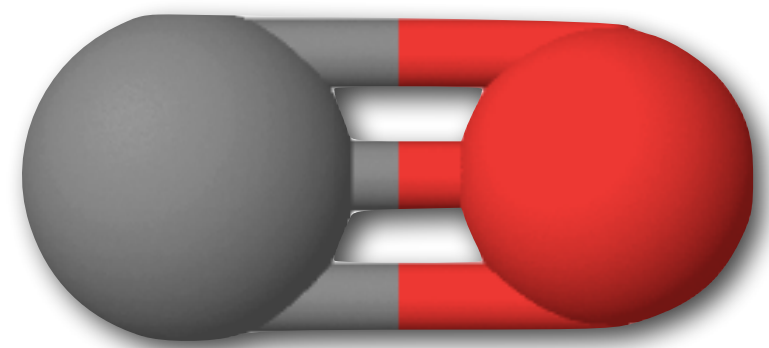


MICHELINE B. SOLEY

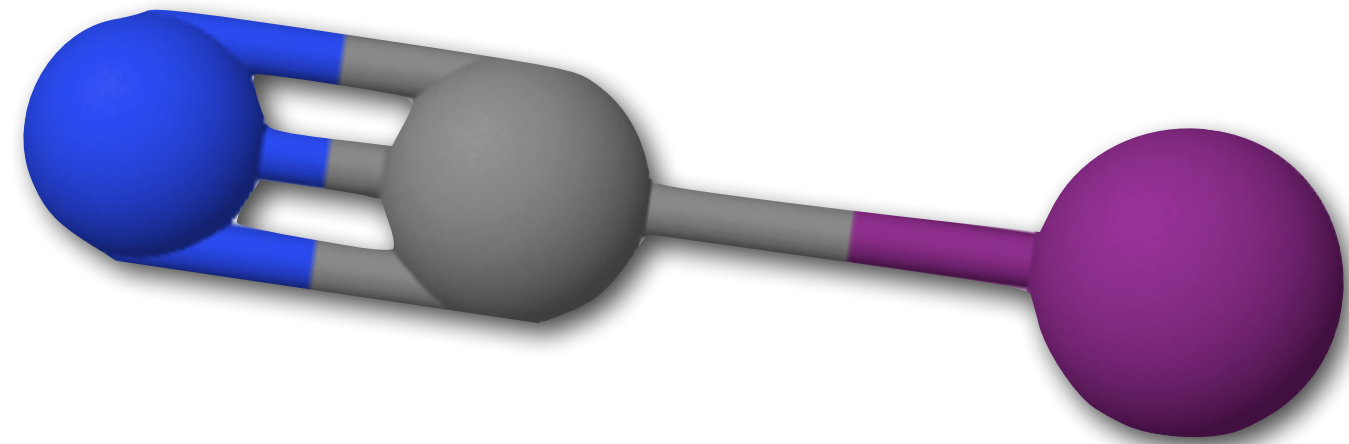
DEPT. OF CHEMISTRY AND DEPT. OF PHYSICS-AFFILIATE,
UNIVERSITY OF WISCONSIN-MADISON
CYBERTRAINING WORKSHOP 2023

COMPUTATIONAL COST OF QUANTUM DYNAMICS AND OPTIMIZATION

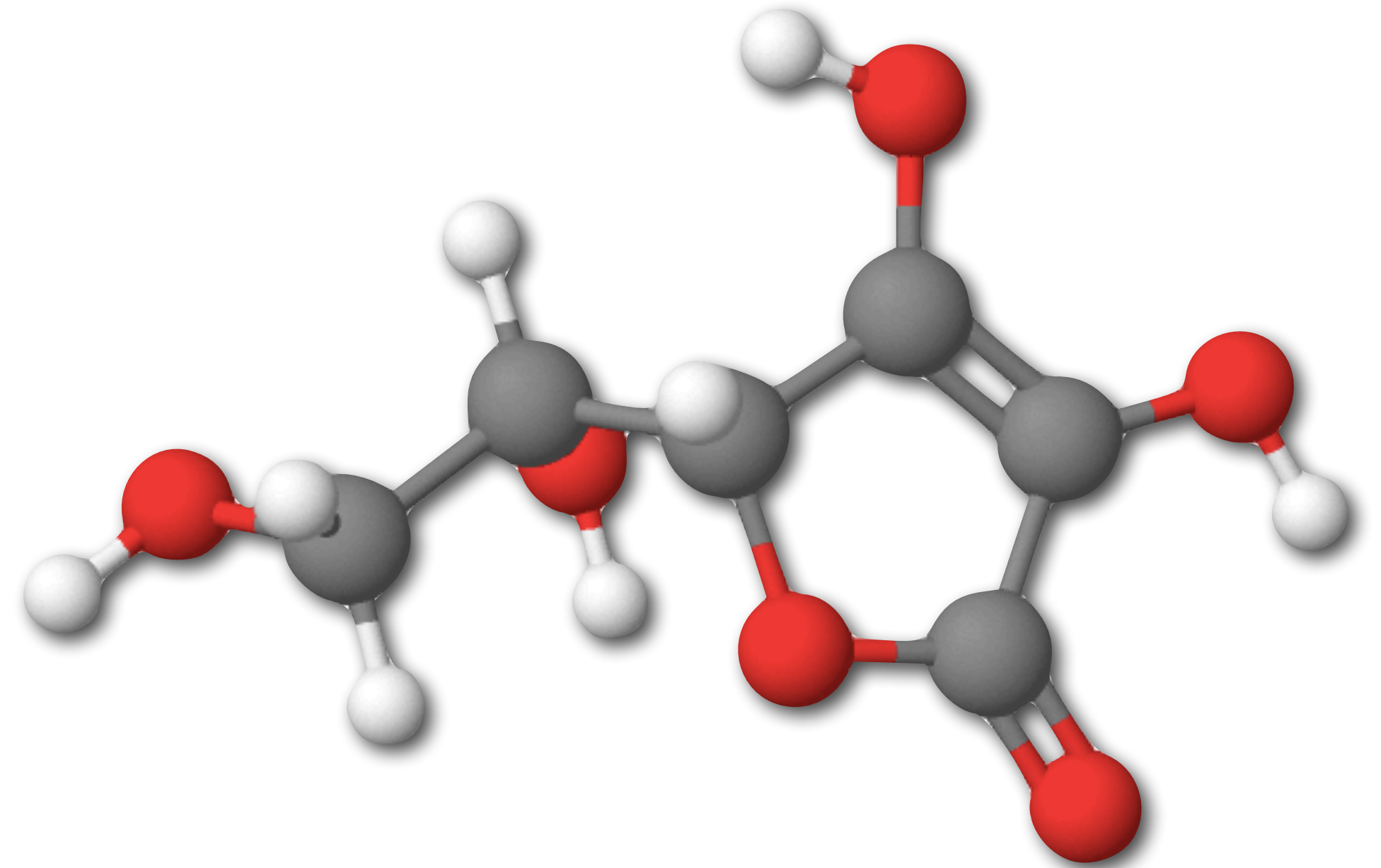
Many classical computer approaches to quantum mechanics simulations become computationally intractable for large molecular systems.



2 Atoms:
2 KB



3 Atoms:
100 MB



20 Atoms:
 10^{119} TB

REDUCING COMPUTATIONAL COST WITH TENSOR TRAINS

Original Image



100%

N. Lyu*, E. Mulvihill*, **Micheline B. Soley**, E. Geva, V. S. Batista, (2023) JCTC, in press.
T. H. Kyaw*, **Micheline B. Soley**,* B. Allen, P. Bergold, C. Sun, V. S. Batista, A. Aspuru-Guzik, (2022) arXiv:2208.10470v1.
Micheline B. Soley,* P. E. Videla,* E. T. J. Nibbering, V. S. Batista, J. Phys. Chem. Lett., 13 (2022) 8354.
Micheline B. Soley, P. Bergold, A. A. Gorodetsky, V. S. Batista, JCTC, 18 (2022) 25.

Tensor Train



15%

Y. Wang, E. Mulvihill, Z. Hu, N. Lyu, S. Shivpuje, Y. Liu, **Micheline B. Soley**, E. Geva, V. S. Batista, S. Kais, 2022, arXiv:2209.04956.
N. Lyu, **Micheline B. Soley**, V. S. Batista, JCTC, 18 (2022) 3327.
Micheline B. Soley, P. Bergold, V. S. Batista, JCTC 17 (2021) 3280.
B. N. Khoromskij, Constr. Approx. 34 (2011) 257.
I. Oseledets, E. Tyryshnikov, Linear Algebra Appl. 432 (2010) 70.
J. C. Napp, et al. Phys. Rev. X 12 (2022) 021021.

REDUCING COMPUTATIONAL COST WITH TENSOR TRAINS

co tt_image.ipynb ☆
File Edit View Insert Runtime Tools Help Last edited on June 18

+ Code + Text

Image Compression

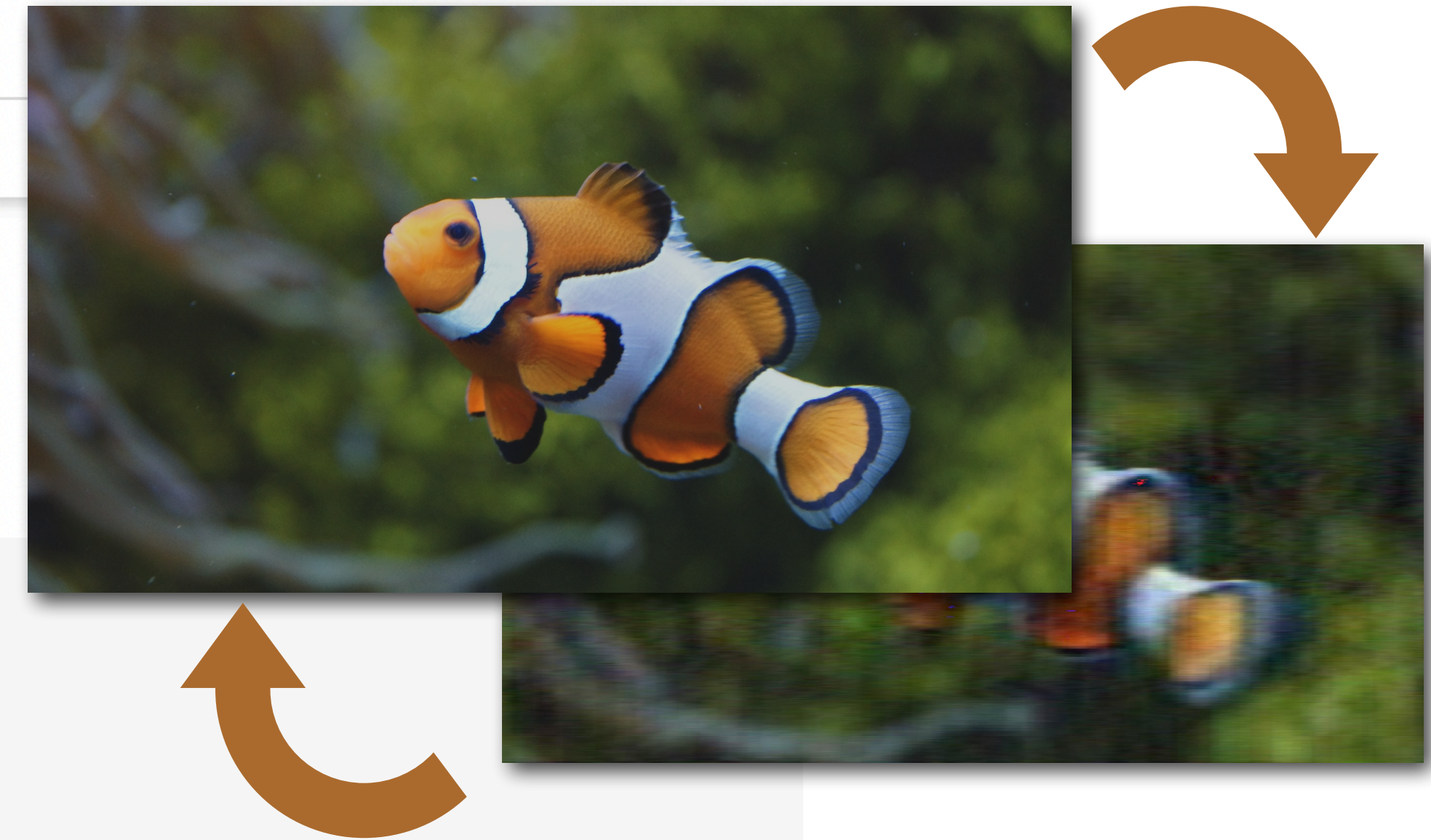
global data global paddedflatdata

```
[ ] # Parameters, image data
eps=1e-2
eps_qttcross=1e-2000
rma=300# 70 # 300 good for quantics, 70 good for TT
rma_qttcross=1e100
rows,columns=img.size
datasize=rows*columns*3
print("Original Size: ",datasize)
img.load()
data=np.asarray(img).reshape((columns,rows*3))
flatdata=np.reshape(data,columns*rows*3)
```

Original Size: 4320000

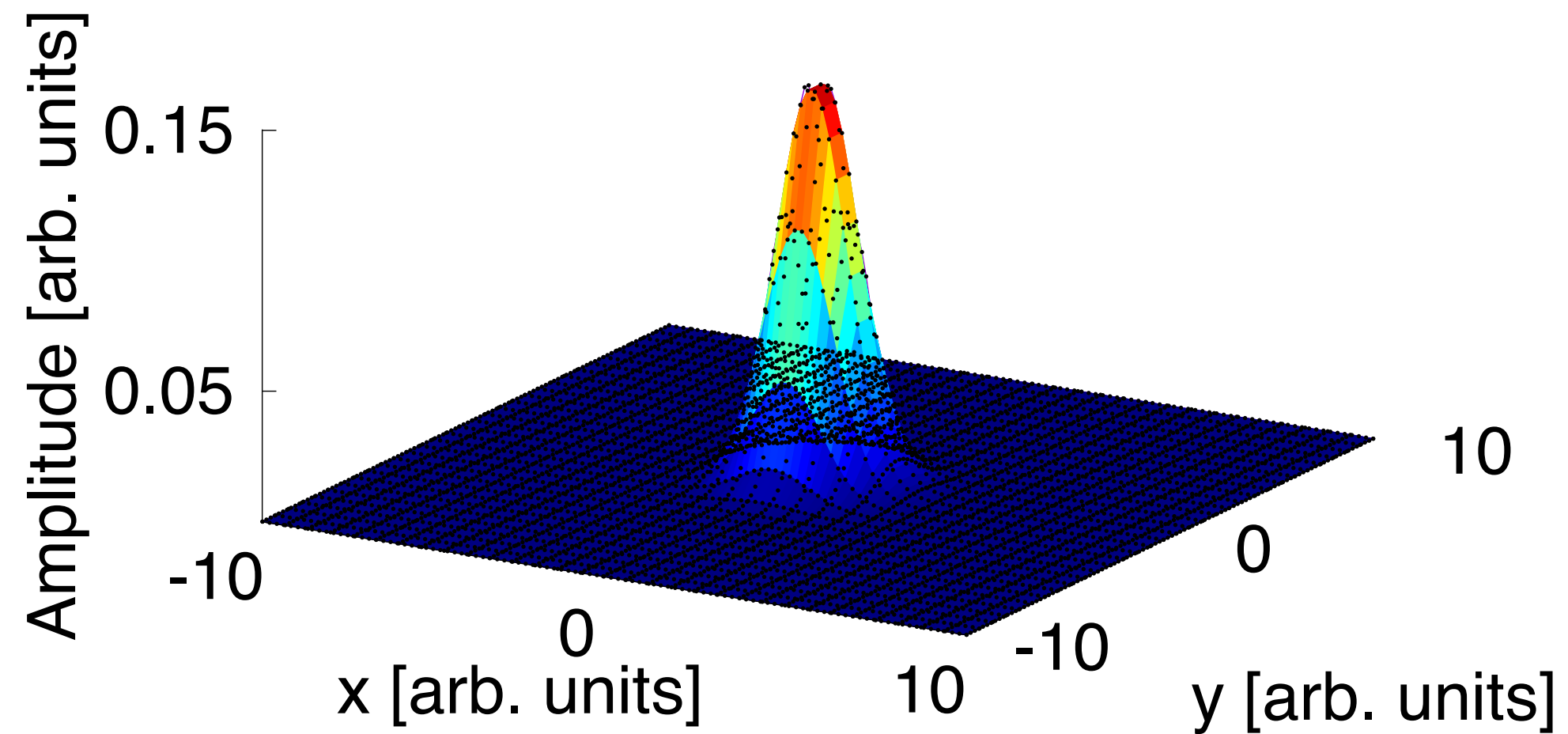
```
[ ] # TT Image

tt_data=tt.tensor(data,eps)
tt_data=tt_data.round(eps,rma)
ttsize=np.size(tt_data.core)
tt_reconstructed_data=np.round(tt_data.full()).reshape((columns,rows,3))
tt_reconstructed_img=Image.fromarray(np.uint8(tt_reconstructed_data))
```

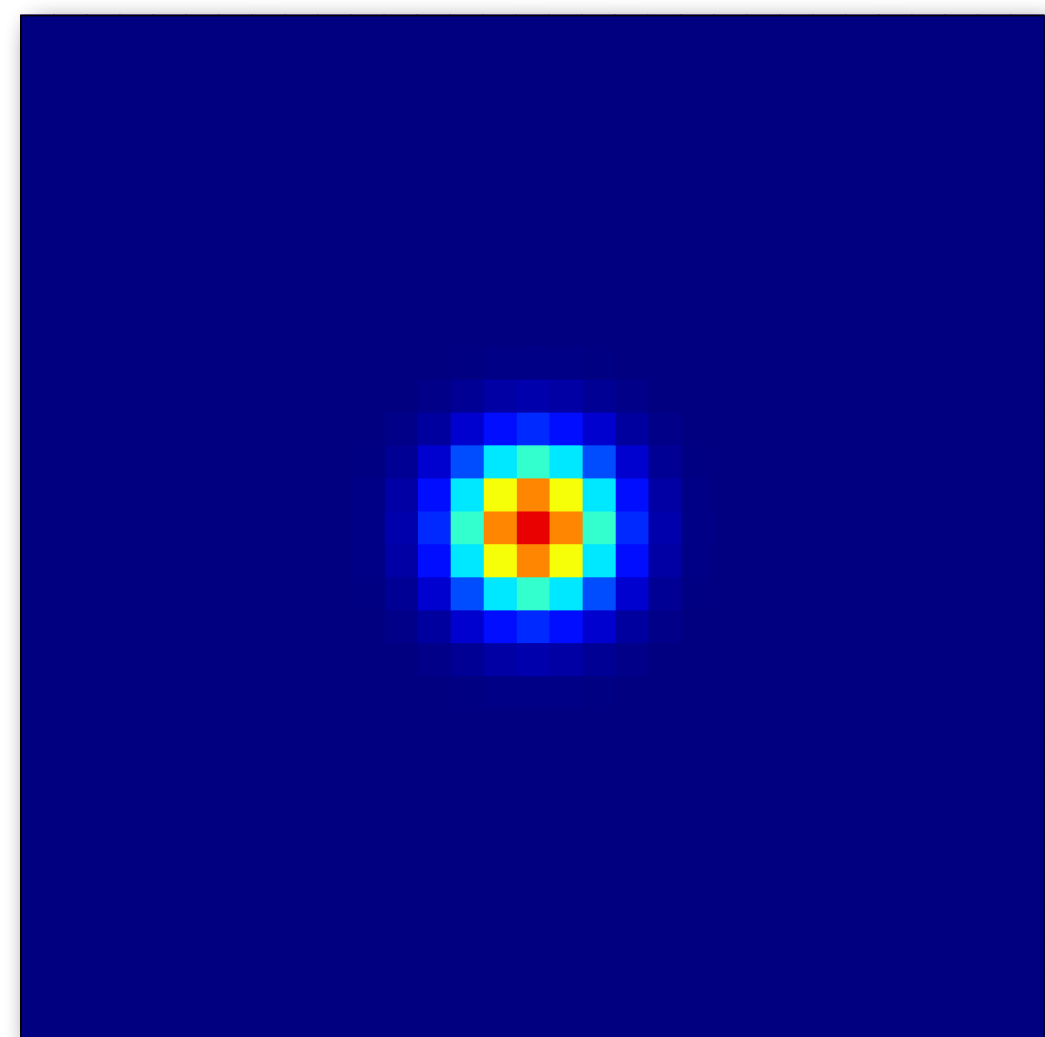


2D Gaussian

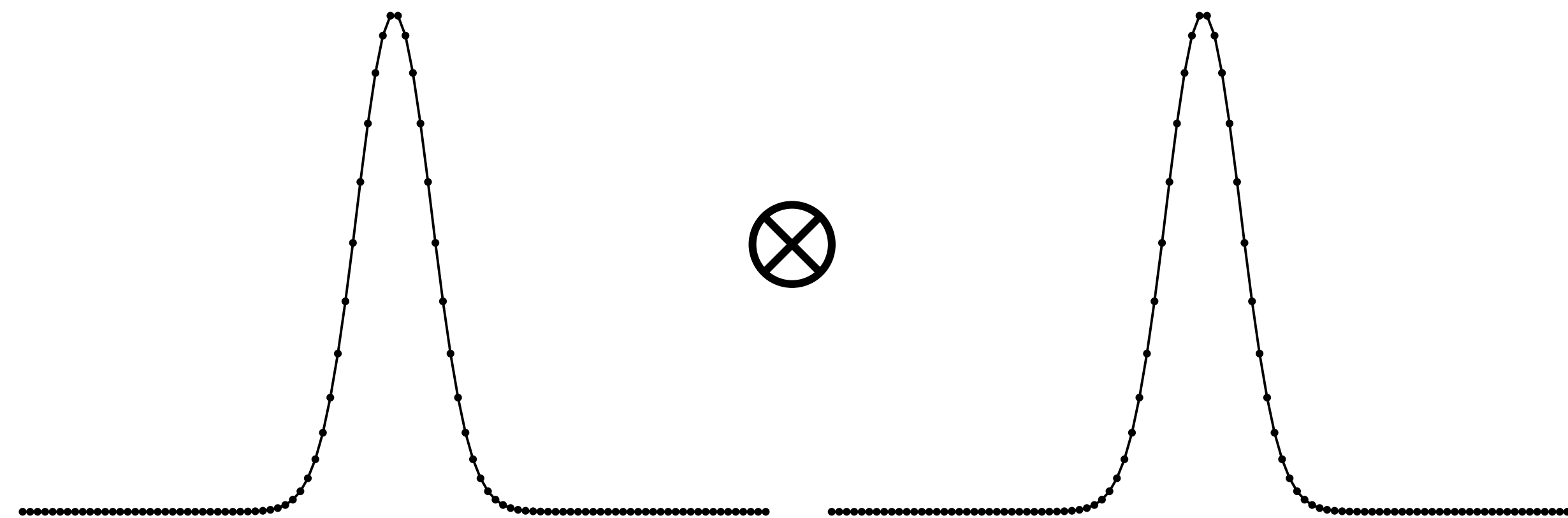
2D TENSOR TRAIN (RANK 1)



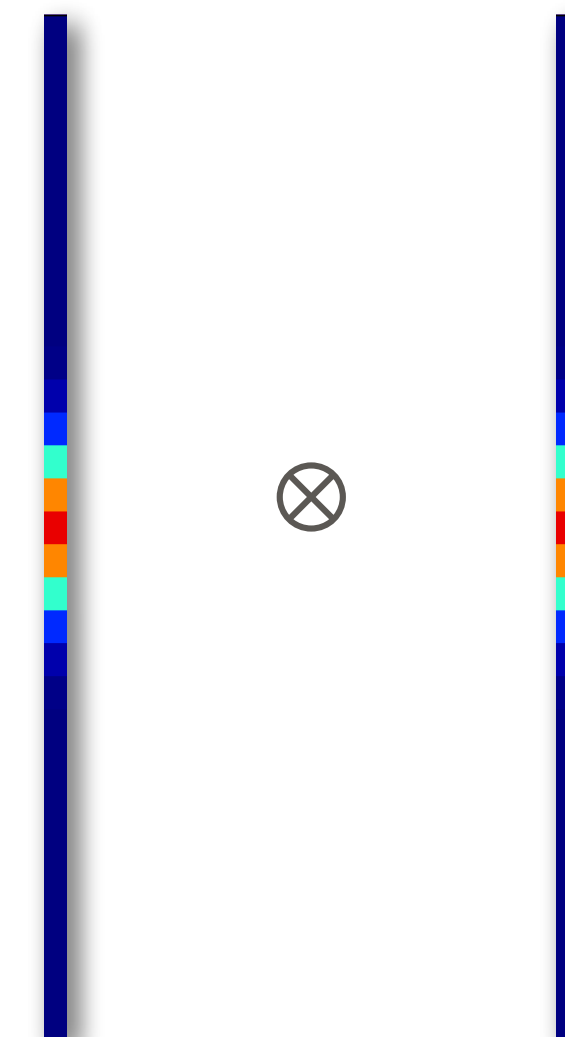
$$\Psi(x, y) = \frac{1}{2\pi} \exp\left(-\frac{x^2}{2} - \frac{y^2}{2}\right)$$



vs.



$$\begin{aligned} \psi(x)\psi(y) &= \left(\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)\right) \left(\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{y^2}{2}\right)\right) \\ &= \sum_{\alpha=1}^r \psi_1(1, x, \alpha) \psi_2(\alpha, y, 1) \end{aligned}$$

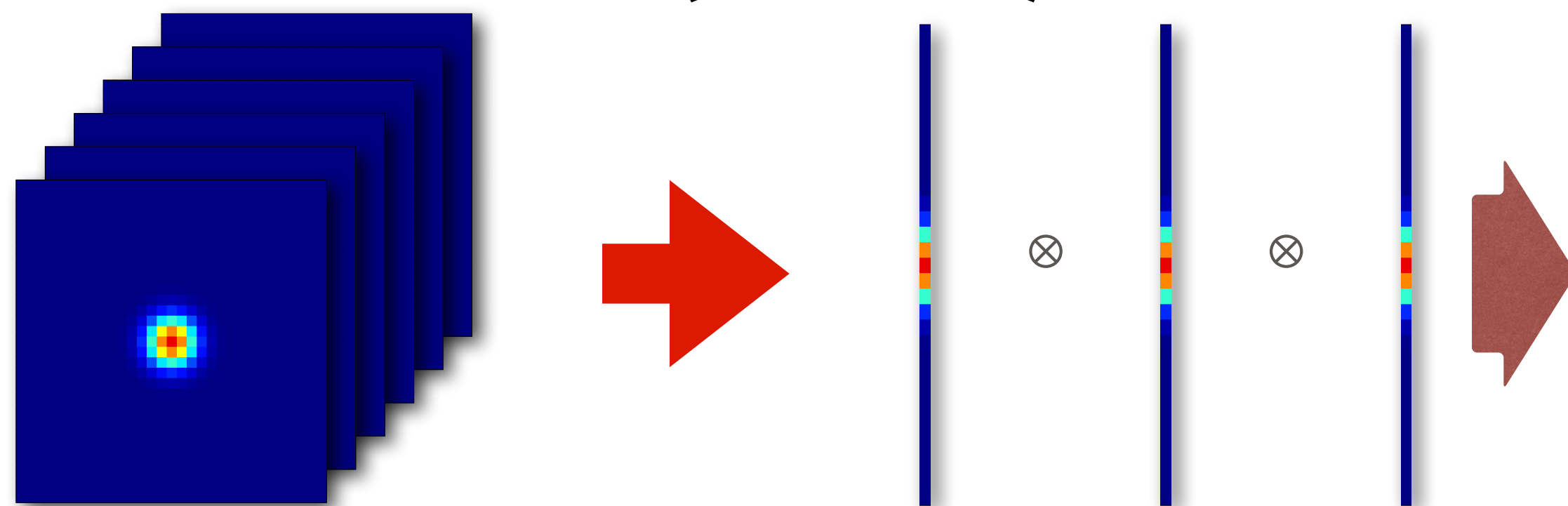


\otimes

Cost of evaluating reduces to

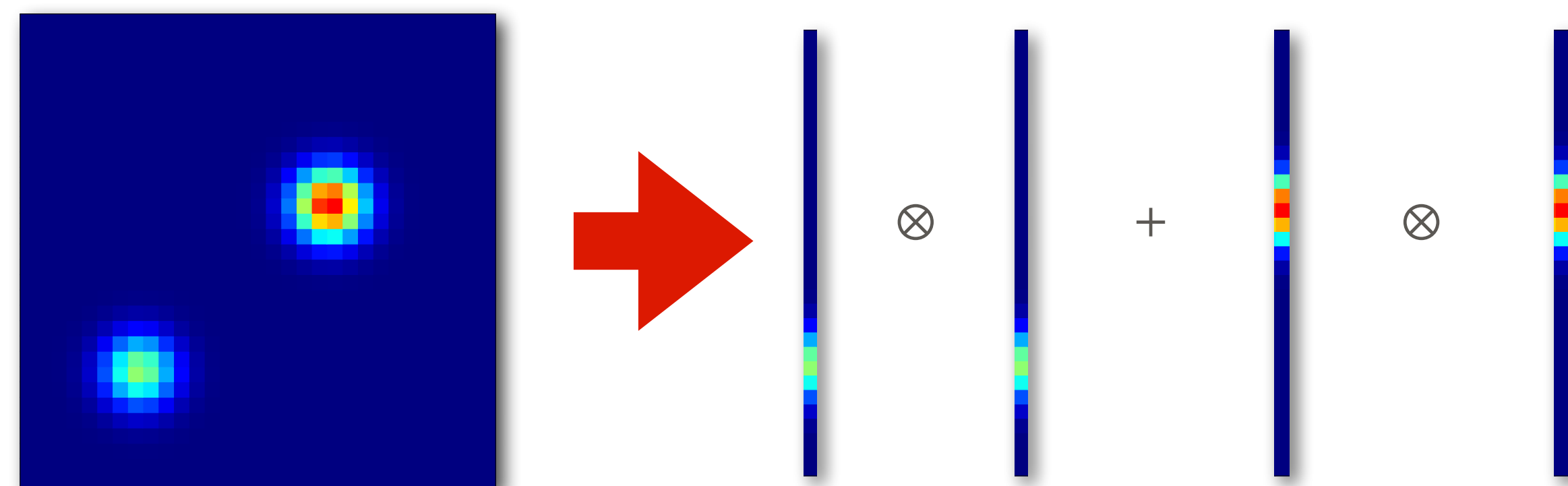
$$n^2 \rightarrow 2n$$

ND TENSOR TRAIN (RANK 1)



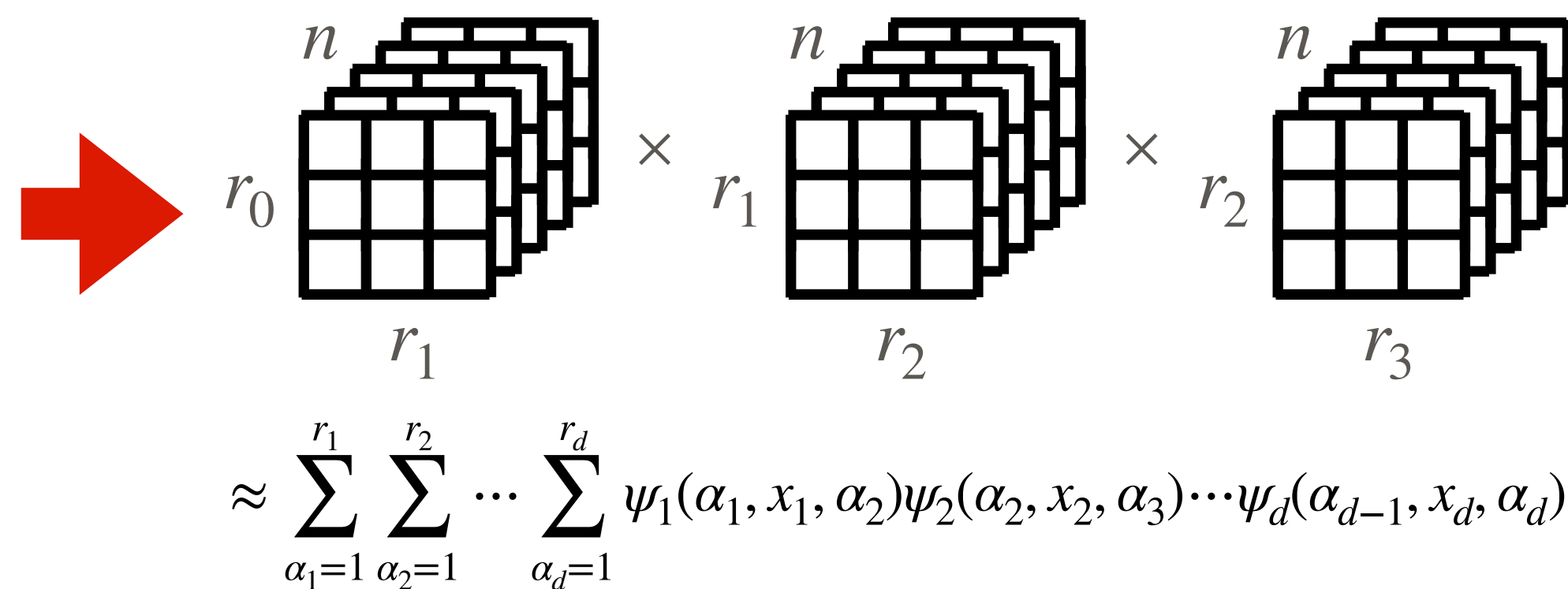
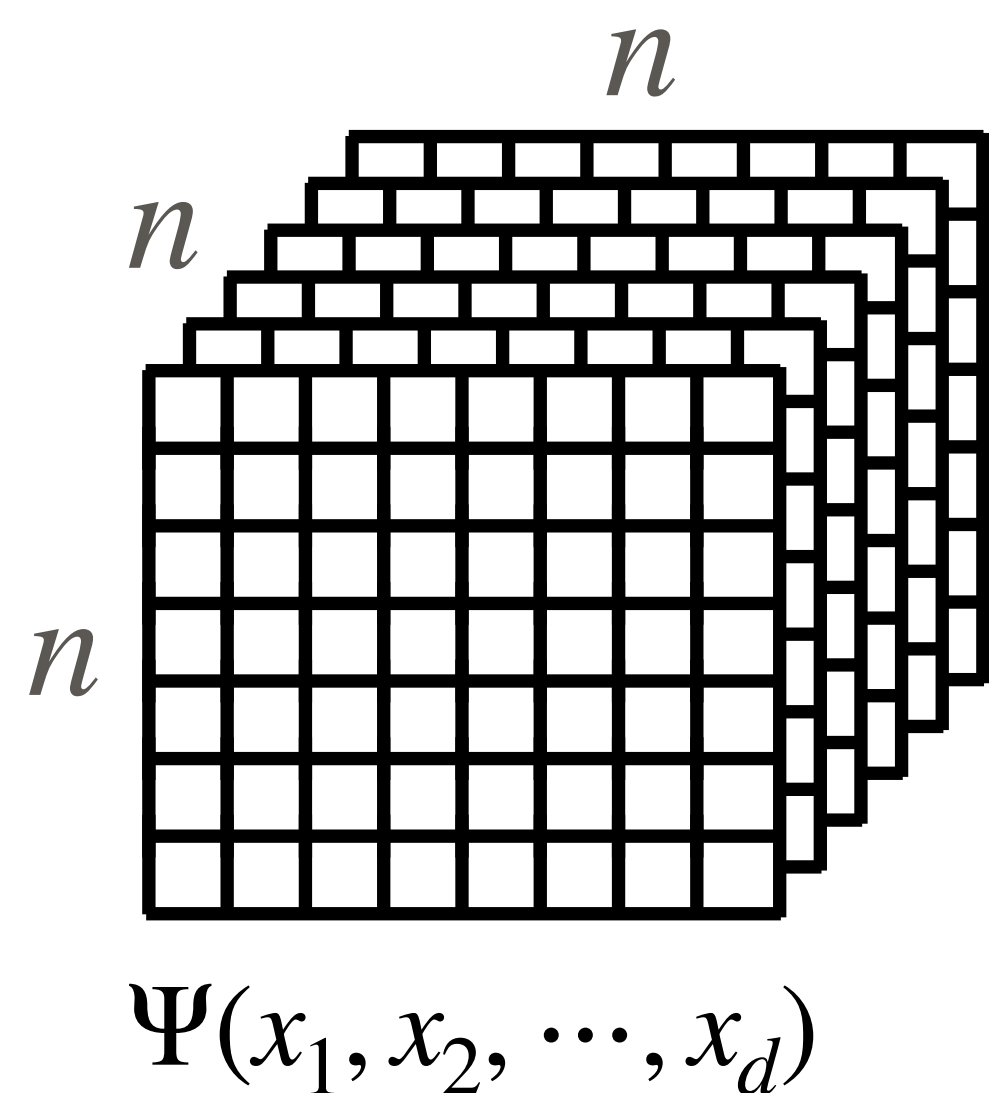
Cost Reduction: $n^3 \rightarrow 3n$

2D TENSOR TRAIN (RANK 2)



Cost Reduction: $n^2 \rightarrow 2 \times 2n$

ND TENSOR TRAIN (RANK R)



Cost Reduction
 $n^d \rightarrow dnr^2$

M.-L. Li, K. S. Candan, M. L. Salino, "GTT: Guiding the tensor train decomposition." International Conference on Similarity Search and Applications. Springer, Cham, 2020.
 I. Oseledets, E. Tyryshnikov, Linear Algebra Appl. 432 (2010) 70.

TT-TOOLBOX



tt_tutorial_redux.ipynb ☆

File Edit View Insert Runtime Tools Help [Last edited on June 18](#)

Comment

Share



M

Table of contents

Import Libraries

Hands-On Tutorial on Tensor Trains

What is a tensor train?

Basic Routines

tt_tensor class

tt_matrix class

Advanced functions

1. Linear System Solver

2. Eigenvalue Solver

3. Cross Approximation of a Black-Box Function

3.1. Fast Evaluation

3.2. Integrals

3.3. Tensor Inversion

4. Minimum of a Function

4.1. Example 1: Minimize 4-d Rosenbrock function on a

+ Code + Text

Connect



▼ Hands-On Tutorial on Tensor Trains

The goal of this tutorial is to get a quick start into the [TT-Toolbox](#) for fast multilinear algebra computations. Here, we introduce the basic routines for multidimensional array operations in TT-format with examples, including a presentation based on the quick start document developed by Ivan Oseledets, Sergey Dolgov, Vladimir Kazeev, Olga Lebedeva, Thomas Mach, and developments at Yale by the Batista group.

▼ What is a tensor train?

The TT-format of a tensor \mathbf{A} is defined, as follows:

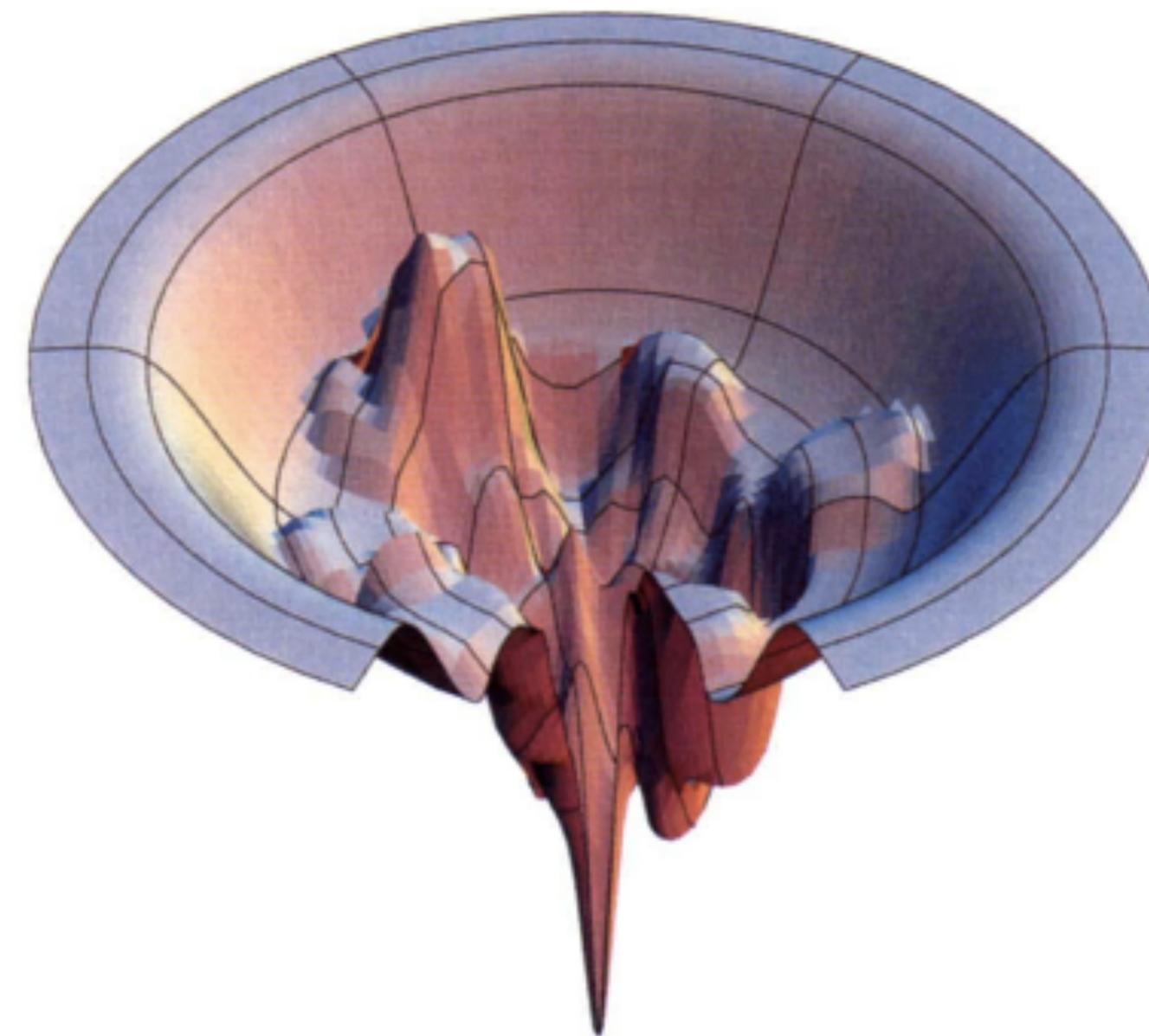
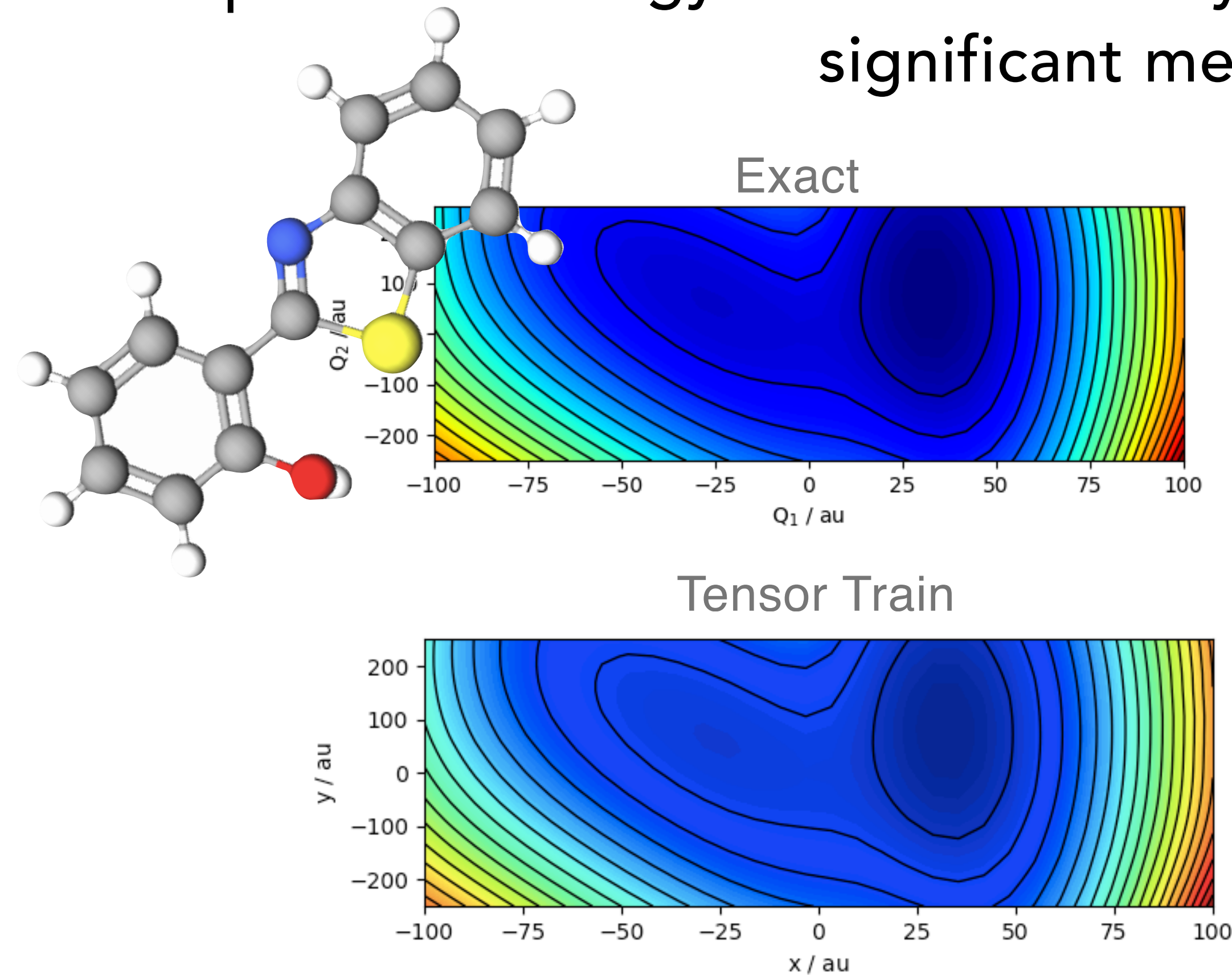
$$A(l_1, l_2, \dots, l_d) = \sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_{d-1}=1}^{r_{d-1}} G_1(l_1, \alpha_1) G_2(\alpha_1, l_2, \alpha_2) \cdots G_d(\alpha_{d-1}, l_d),$$

or more concisely $A(l_1, l_2, \dots, l_d) = G_1(l_1) \times G_2(l_2) \times \cdots \times G_d(l_d)$, where $G_k(l_k)$ are $r_{k-1} \times r_k$ matrices, and $r_0 = r_d = 1$, as represented in the following diagram:



TENSOR TRAINS FOR CHEMISTRY

For potential energy surfaces of many chemical systems, tensor trains offer significant memory savings.



We take advantage of this property for quantum dynamics and global optimization of molecular systems in high dimensionality.

N. Lyu*, E. Mulvihill*, **Micheline B. Soley**, E. Geva, V. S. Batista, JCTC, 19 (2023) 1111.

Micheline B. Soley*, P. E. Videla,* E. T. J. Nibbering, V. S. Batista, J. Phys. Chem. Lett., 13 (2022) 8354.

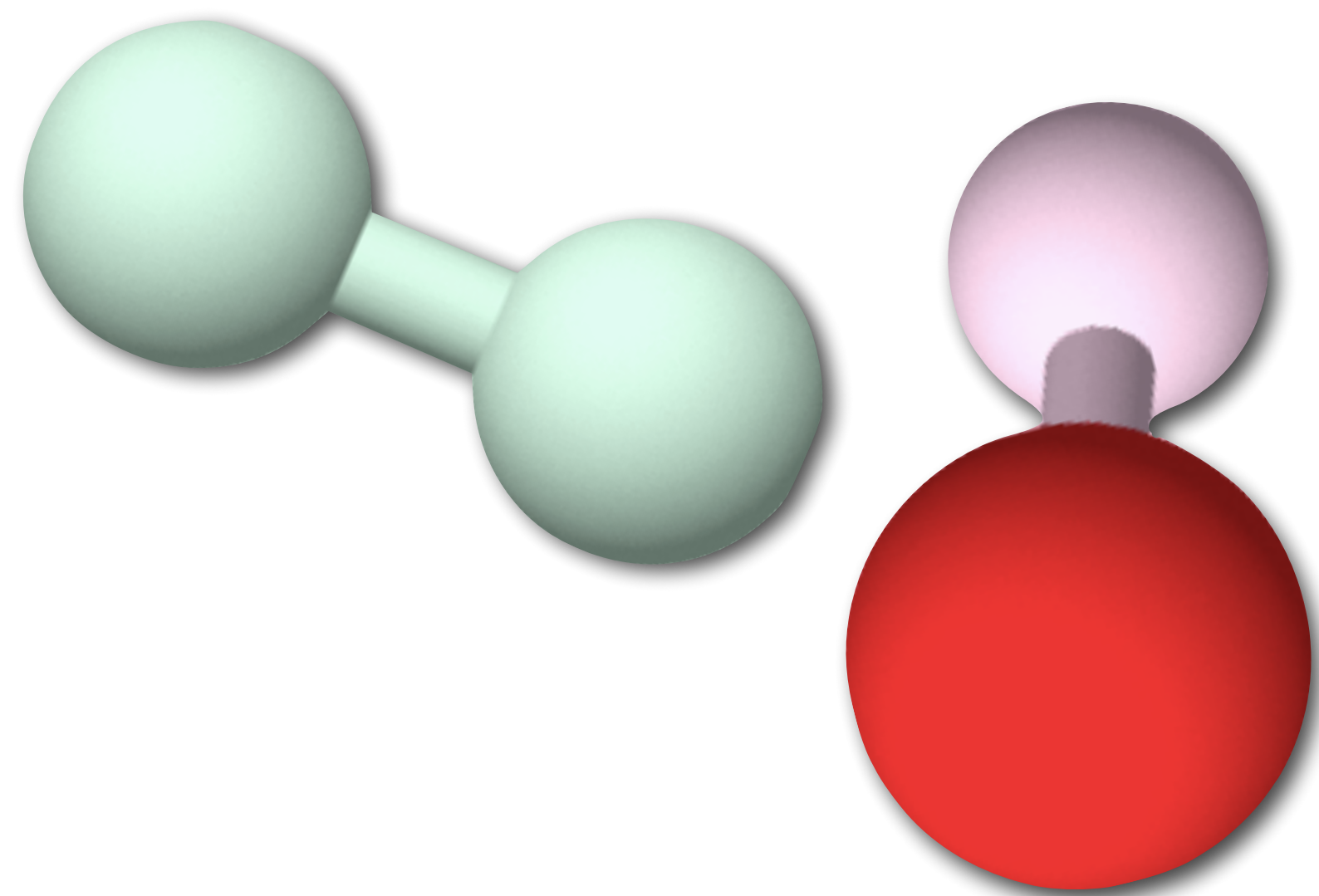
Micheline B. Soley, P. Bergold, A. A. Gorodetsky, V. S. Batista, JCTC, 18 (2022) 25.

N. Lyu, **Micheline B. Soley**, V. S. Batista, JCTC, 18 (2022) 3327.

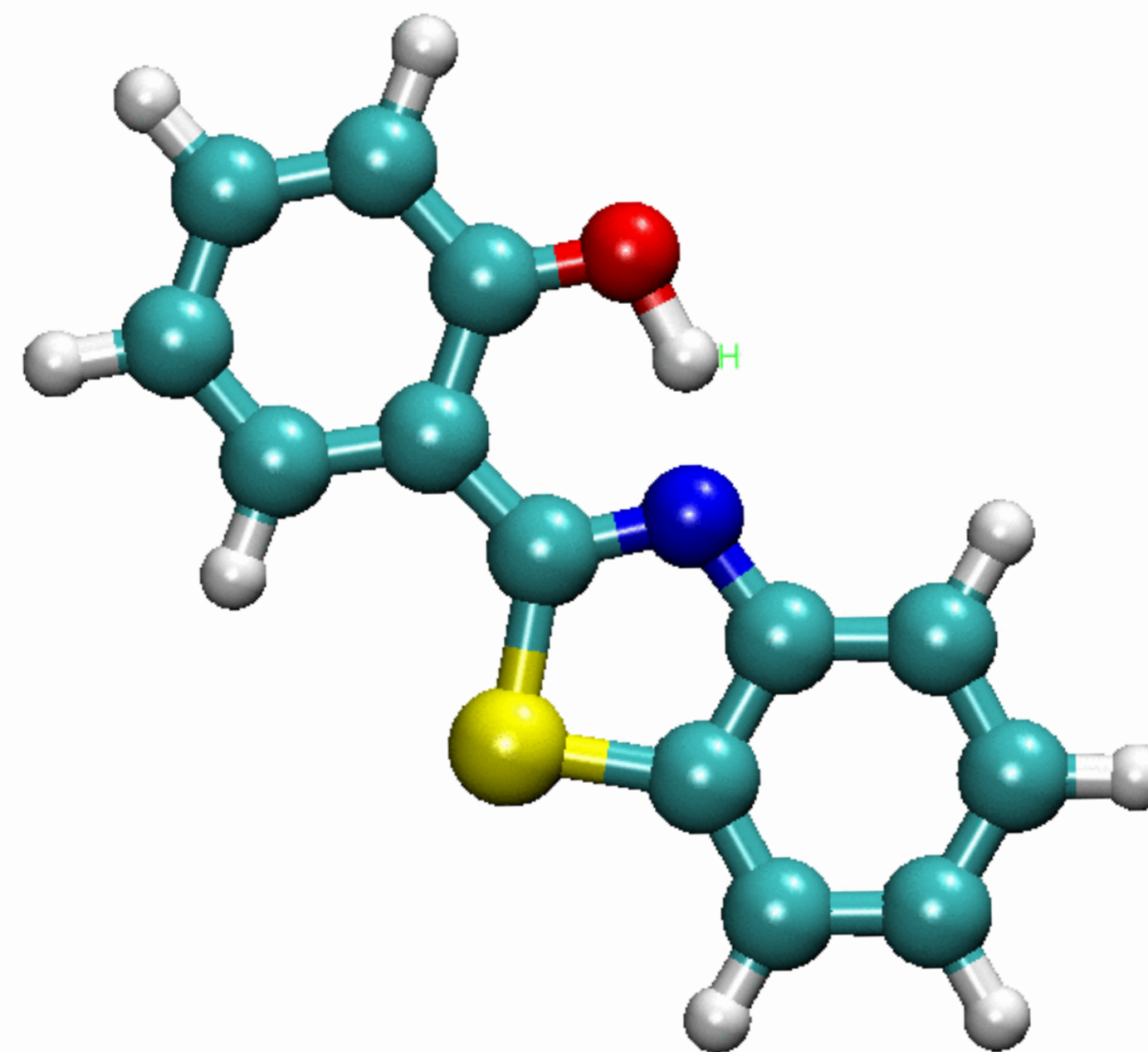
Micheline B. Soley, P. Bergold, V. S. Batista, JCTC 17 (2021) 3280.

APPLICATION 1: HIGHLY-MULTIDIMENSIONAL QUANTUM DYNAMICS OF MOLECULES WITH TT-SOFT

Largest System Investigable with Standard Fixed-Grid SOFT Dynamics



Tensor-Train Split-Operator Fourier Transform (TT-SOFT) Dynamics



J. A. Fleck Jr., A. Steiger, J. Comput. Phys. 47 (1982) 412.
S. M. Greene, V. S. Batista, JCTC 13 (2017) 4034.

Micheline B. Soley,* P. E. Videla,* E. T. J. Nibbering, V. S. Batista, (2022) J. Phys. Chem. Lett., 18 (2022) 8254.

APPLICATION 1: HIGHLY-MULTIDIMENSIONAL QUANTUM DYNAMICS OF MOLECULES WITH TT-SOFT

The screenshot shows the GitHub interface for the repository 'michelinesoley / HBT'. The repository is public and has 1 star, 0 forks, and 0 notifications. The main branch is 'main'. The repository contains a commit by Pablo Videla on April 6, 2022, with 32 commits. The commit message is 'delete unnecessary files'. The repository contains the following files and folders:

File/Folder	Commit Message	Time Ago
DynamicsCodes	Fixed norm.gpt error	2 years ago
PotentialsCodes	delete unnecessary files	last year
README.md	Update README.md	2 years ago
documentation.pdf	Added documentation	2 years ago

The README.md file is visible and contains the following text:

HBT

TT-SOFT code for determination of UV pump/X-ray probe UV spectra and propagation of HBT with fully quantum treatment of all 69 degrees of freedom.

The right sidebar contains the following information:

About

TT-SOFT code for fully quantum dynamics and determination of UV pump/X-ray probe spectra

- Readme
- 1 star
- 3 watching
- 0 forks

Report repository

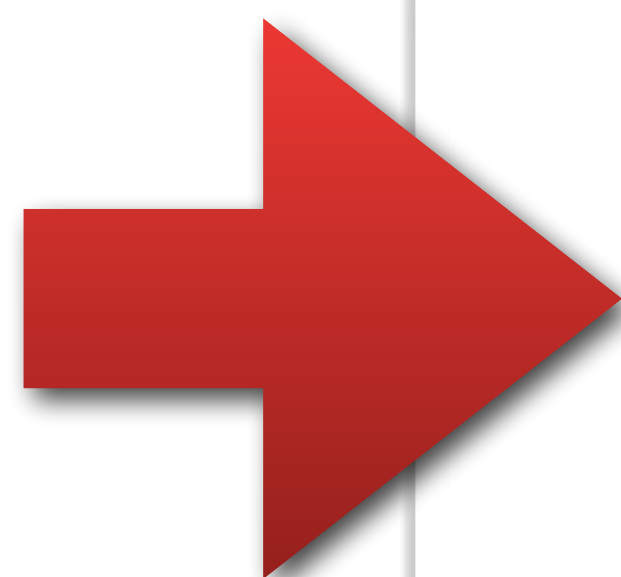
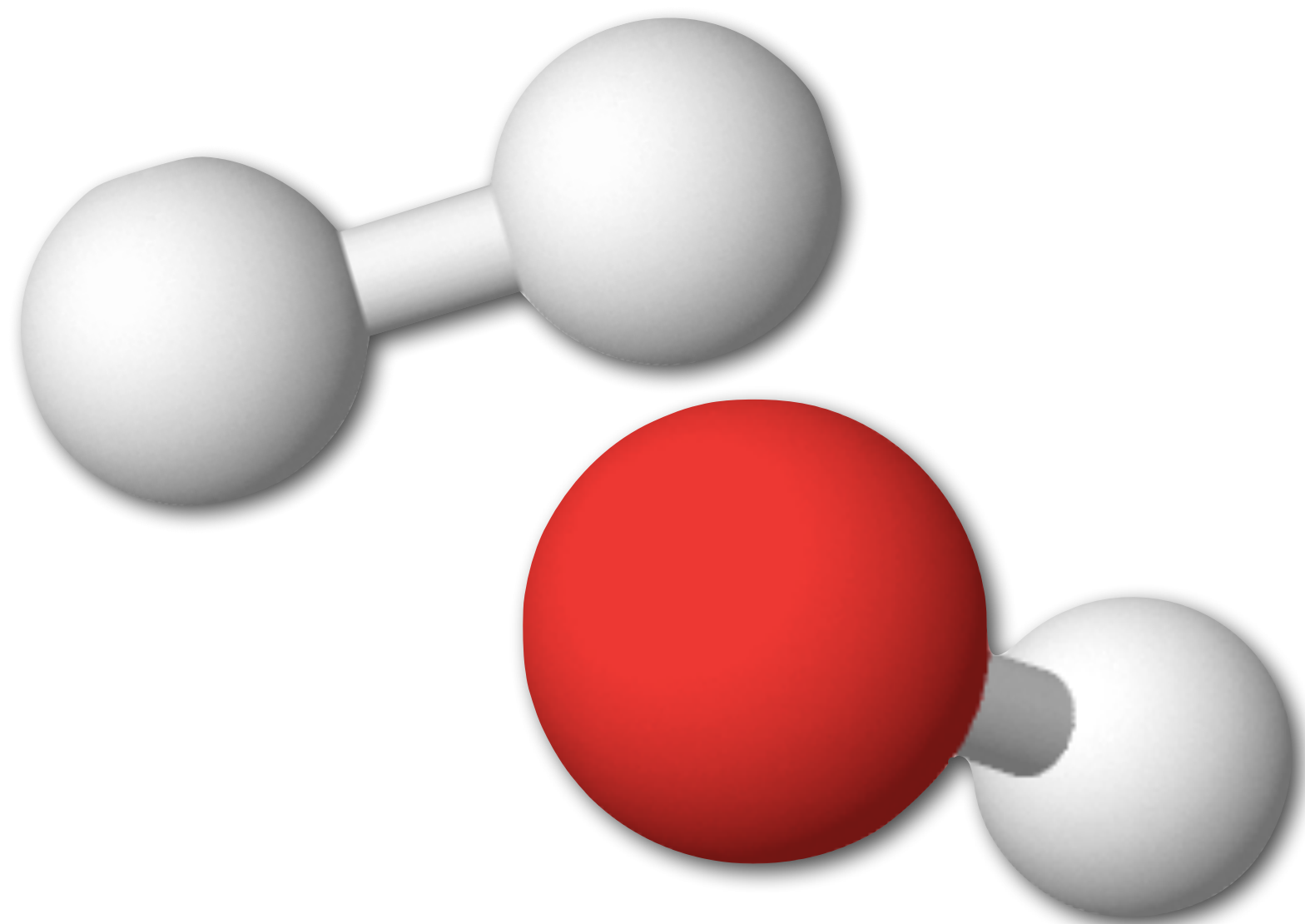
Releases

No releases published

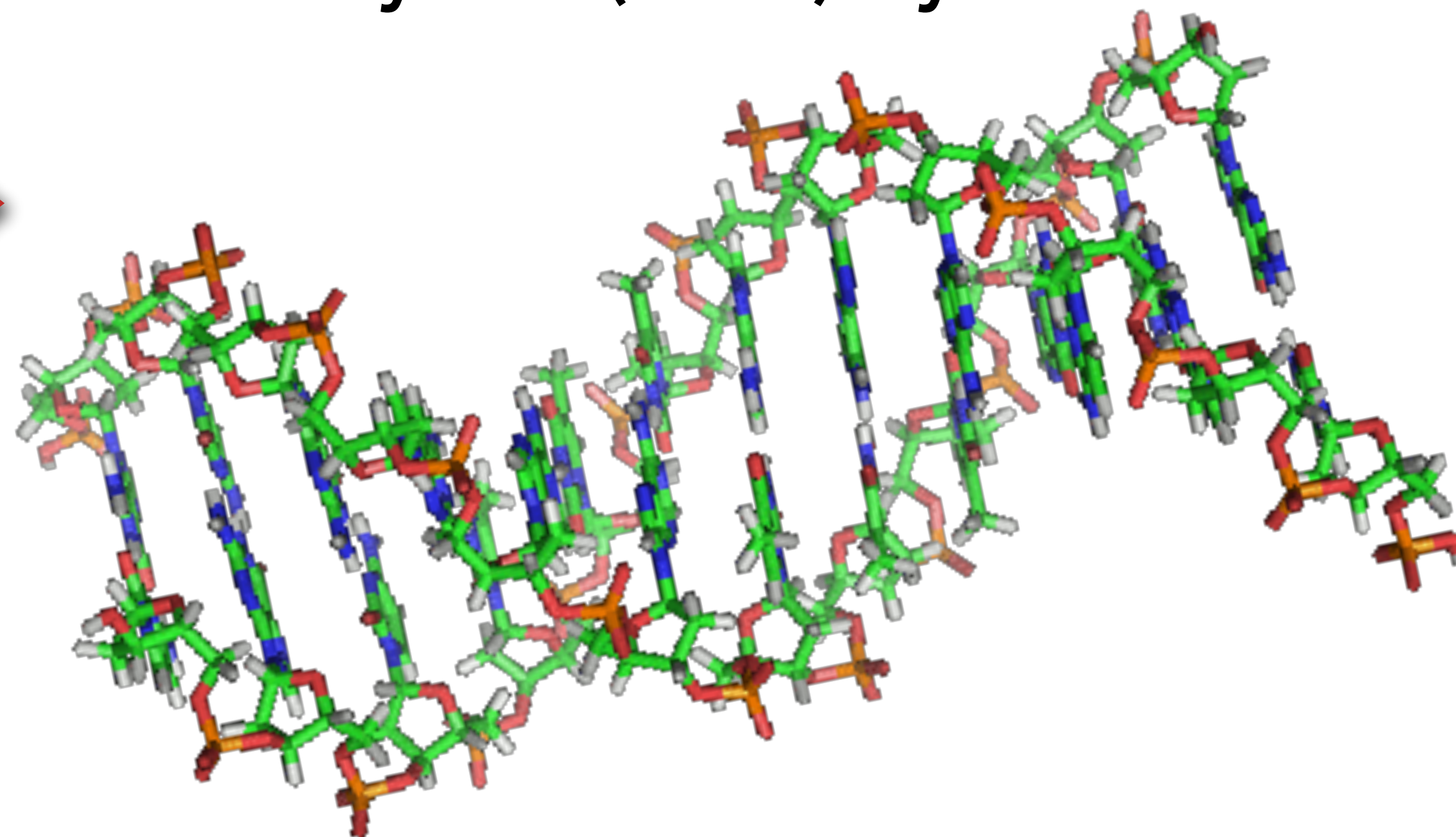
Packages

APPLICATION 2: HIGHLY-MULTIDIMENSIONAL QUANTUM DYNAMICS OF MOLECULES WITH TT-CHEBYHSEV

Largest System Investigable with Standard Chebyshev Dynamics



Functional Tensor-Train
Chebyshev (FTTC) Dynamics



M. T. Cvitaš, S. C. Althorpe, J. Chem. Phys. 139 (2013) 064307.
E. M. Goldfield, S. K. Gray, J. Chem. Phys. 117 (2002) 1604.
H. Tal-Ezer, R. Kosloff, J. Chem. Phys. 81 (1984) 3967.

[Micheline B. Soley, P. Bergold, A. A. Gorodetsky, V. S. Batista, JCTC, 18 \(2022\) 25.](#)

APPLICATION 2: HIGHLY-MULTIDIMENSIONAL QUANTUM DYNAMICS OF MOLECULES WITH TT-CHEBYHSEV

The screenshot shows the GitHub interface for the repository 'FTTC' by 'michelinesoley'. The repository is public and has 1 star, 0 forks, and 0 notifications. The main branch is 'main' with 1 branch and 0 tags. The repository contains a README.md file and several folders: Comparisons, FTTC, FTTCPython, and TT. The FTTC folder contains updated FTTC nomenclature, and the TT folder also contains updated FTTC nomenclature. The FTTCPython folder contains Python FTTC code. The README.md file contains Python FTTC code. The repository is described as a simulation of chemical dynamics in high dimensionality with low-rank functional tensor-train Chebyshev (FTTC) quantum dynamics.

Product Solutions Open Source Pricing Search Sign in Sign up

michelinesoley / FTTC Public Notifications Fork 0 Star 1

Code Issues Pull requests Actions Projects Security Insights

main 1 branch 0 tags Go to file Code

michelinesoley Update CodeDocumentation.pdf f84b73e on Feb 28, 2022 14 commits

Comparisons	Updated FTTC nomenclature	2 years ago
FTTC	Updated FTTC nomenclature	2 years ago
FTTCPython	Python FTTC	last year
TT	Updated FTTC nomenclature	2 years ago
CodeDocumentation.pdf	Update CodeDocumentation.pdf	last year
README.md	Python FTTC	last year

README.md

About
Simulation of chemical dynamics in high dimensionality with low-rank functional tensor-train Chebyshev (FTTC) quantum dynamics
Readme
1 star
1 watching
0 forks
Report repository

Releases
No releases published

APPLICATION 3: HIGHLY-MULTIDIMENSIONAL OPTIMIZATION FOR MOLECULAR SYSTEMS WITH IPA

IPA

Shor's Algorithm in Quantum Computing

$$21 = 7 \times 3$$



N=706851632784678312266808500466226101921669464385547527413569690763875796535401970800670478642
89788827197635974448256235924525587148173962379557986835363631441965874443074041408313128692006
36395978572420256222807383699910685528569264451954359616541691212293741597127675540821848168
9308953580556662876975166940955341296487306901250434774718332134979356605095177438931303780173
137245526404005011234493448949129910141059289267762650584326974113659206112726332997462438687870
05441738484521934554593134411734055431434314113765065278125412502358263711235619202289345940500
4551457448831654483376775960882508814275718823140983832110651611747166615832389084594281762
9104852608687052760339612206092247760952529265134188198368246118801152700133324524885486480
997674192024201965727093108011252558162000981820812899621145140001270371422310651075
867403455271862309293906286499648242242986288787211529045105741199515713370561223649123085
7578039871731184009403136764320837641102143677316631099872153409711911800671914907004104990
1366428499722366877530724942372323524394104133335987147459111011010339652063399874368946520524
420895339479018793070897634840285234432774712319992778101223105790841422230843219659893780948
6178812793382703966190332059501965113162510913181163220503118956964945986023783919248741071072
9622697126269234268792582220928818442356351131011595822918184991428775345610409787261221110792
047573908774977452473950325112788755615951431155928363904800199351760947325479155433825636657097
387649882679335732149029616430970902463176875116864116761211149805195358784563373269368287844
69908141667270480130190115828443118723318199421154083614018417070261173839865411330806649094794
36749767875171215421913186145511849021129611185712110296587628275367562891236632051499981
7538397565992411112761132241112211110721198651771543118900483543794055612391518556384
9416492742208121412493161119741118011370123638184171120320663510359674170467923211831676
33035292671411911307911755210116528091112211111300137114203637051377141516923445674118786787
0190383373511128013948110460911132954314178111131224882169487512573279160687585894629855333685
6047064094291555211626978117925393114016311191209891628750585199171480727149244103302709672451
57900985635244765591191336411083058711364541912544886341889094948465981128593596305804674031386
4942410681616899071251127826331137449781159011521465250447489337455902640728012518915859962093979
227459401522552702162111178411119116111142115460631756231339365887330840484169917017828763051
42071850148474927781191111517631221112021665835862594509151025242869798254964873565372048920491
70025839399556978593608627643113361106611693705353525252656142323642791967328180835792001= 3⁴⁰⁰ ×
11²⁰⁰ × 17²⁰⁰ × 23²⁰⁰ × 41²⁰⁰ × 53²⁰⁰ × 79²⁰⁰ × 101²⁰⁰ × 109²⁰⁰

Micheline B. Soley, P. Bergold, A. A. Gorodetsky, V. S. Batista, JCTC, 18 (2022) 25.

T. H. Kyaw*, Micheline B. Soley,* B. Allen, P. Bergold, C. Sun, V. S. Batista, A. Aspuru-Guzik, (2022) arXiv:2208.10470v1.

P. W. Shor, Proceedings 35th Annual Symposium on Foundations of Computer Science, IEEE (1994) 124.

E. Lucero, et al. Nat. Phys. 8 (2012) 719.

APPLICATION 3: HIGHLY-MULTIDIMENSIONAL OPTIMIZATION FOR MOLECULAR SYSTEMS WITH IPA

```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
import matplotlib.pyplot as plt
import tt

beta=10
dim=2
eps=1.0e-14
rma=3
nsteps=30
d=8
npts=2**d
xmin=-1.5
xmax=2.5

dx=(xmax-xmin)/npts

def gen_1d(mat,e,i,d):
    w=mat
    for j in range(i):
        w=tt.kron(e,w)
    for j in range(d-i-1):
        w=tt.kron(w,e)
    return w

def rhoo(input):
    out=1.+0.*np.sum(input,axis=1)
    return(out);
```

```
import numpy as np
from numpy import zeros,reshape,sqrt,arange,vectorize,extract,int

import matplotlib.pyplot as plt
import tt

import mpmath
from mpmath import mp,mpf,floor,exp,nint

def parameters():
    global dim,eps,num,rmax,nsteps,d,searchspacesize,beta,betaprime
    num=mpf(3*3*11*17*23*41*53*79*101*109)**200
    beta=30
    betaprime=0.5
    dim=1
    eps=1.0e-100
    rmax=100
    nsteps=3
    d=6
    searchspacesize=2**d
    return()

def rhoo(input):
    V=1.0+0*input
    return V

def is_prime(n):
    if n % 2 == 0 and n > 1:
        return False
    return all(n % i for i in range(3,int(sqrt(n))+1,2))

def tto(input,param=None):
    global num,beta
    nevals,dim=input.shape
    out=np.zeros((nevals,))
    for ii in range(nevals):
        a=num-nint(input[ii,0])*floor(num/nint(input[ii,0]))
```

IMPLEMENTATION: TENSOR TRAINS ON GOOGLE COLAB

The low computational cost of these methods allows the codes presented in this session to be run either on personal computers or on a single core on clusters/cloud-based resource. We will run codes today on Google Colaboratory.

Google Colaboratory: https://colab.research.google.com/?utm_source=scs-index#

Colab Python Tutorial: <https://colab.research.google.com/github/cs231n/cs231n.github.io/blob/master/python-colab.ipynb>

IMPLEMENTATION: TENSOR TRAINS ON GOOGLE COLAB

Welcome To Colaboratory

File Edit View Insert Runtime Tools Help

Share Settings M

Table of contents

- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- + Section

+ Code + Text Copy to Drive

Connect

Welcome to Colab!

If you're already familiar with Colab, check out this video to learn about interactive tables, the executed code history view, and the command palette.

3 Cool Google Colab Features

What is Colab?

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

TENSOR TRAINS ON CCR/CLUSTERS

Jupyter Notebook Environments:

<https://ubccr.freshdesk.com/support/solutions/articles/13000073876-custom-jupyter-notebook-environment-setup>

<https://ubccr.freshdesk.com/support/solutions/articles/13000063899-configuring-your-environment-and-private-modules-for-use-in-ondemand>

TENSOR TRAINS ON PERSONAL COMPUTERS

TT-Toolbox: <https://github.com/oseledets/TT-Toolbox>

Compressed Continuous Computation (C3): <https://github.com/goroda/Compressed-Continuous-Computation>

TENSOR TRAINS ON TT-TOOLBOX WITH M1-CHIP MACS

```
conda activate /Users/*****/opt/anaconda3
```

```
conda install numpy
```

```
conda install scipy
```

```
conda install cython
```

```
brew reinstall gcc
```

```
brew unlink open-mpi
```

```
brew reinstall gfortran
```

```
FFLAGS='-fallow-argument-mismatch'
```

```
python setup.py install
```

TENSOR-TRAIN DATA COMPRESSION

In order to understand tensor-train data compression, it is essential to understand the **tensors** on which they are based:

$$\mathcal{A} = [A(i_1, i_2, \dots, i_d)], \quad i_k \in \{1, 2, \dots, n_k\}$$

Dimension D : number of indices

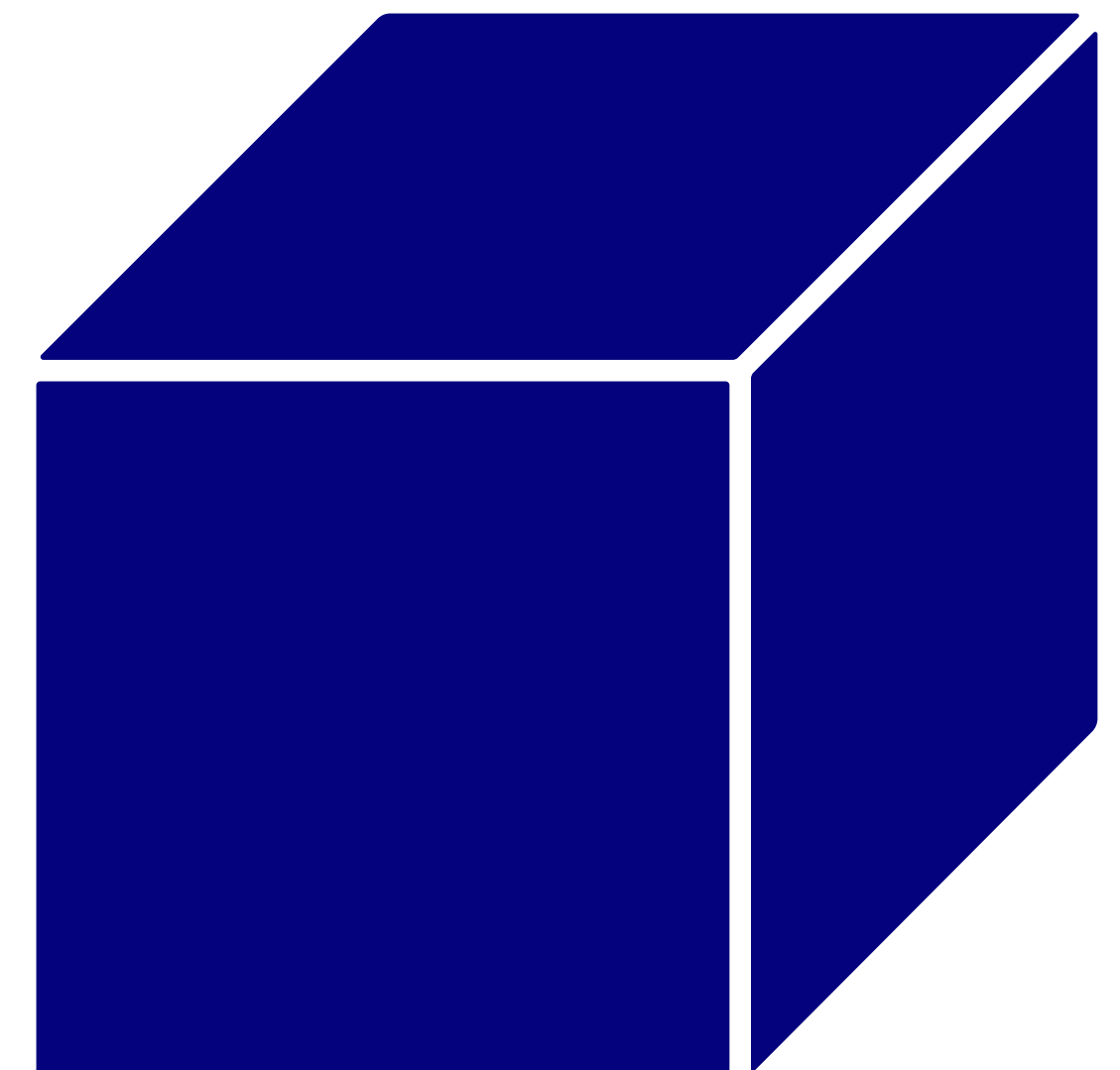
Size: $n_1 \times n_2 \times \dots \times n_D$.

Zeroth-Order (0D) Tensor: Scalar a

First-Order (1D) Tensor: Vector \mathbf{a}

Second-Order (2D) Tensor: Matrix \mathbf{A}

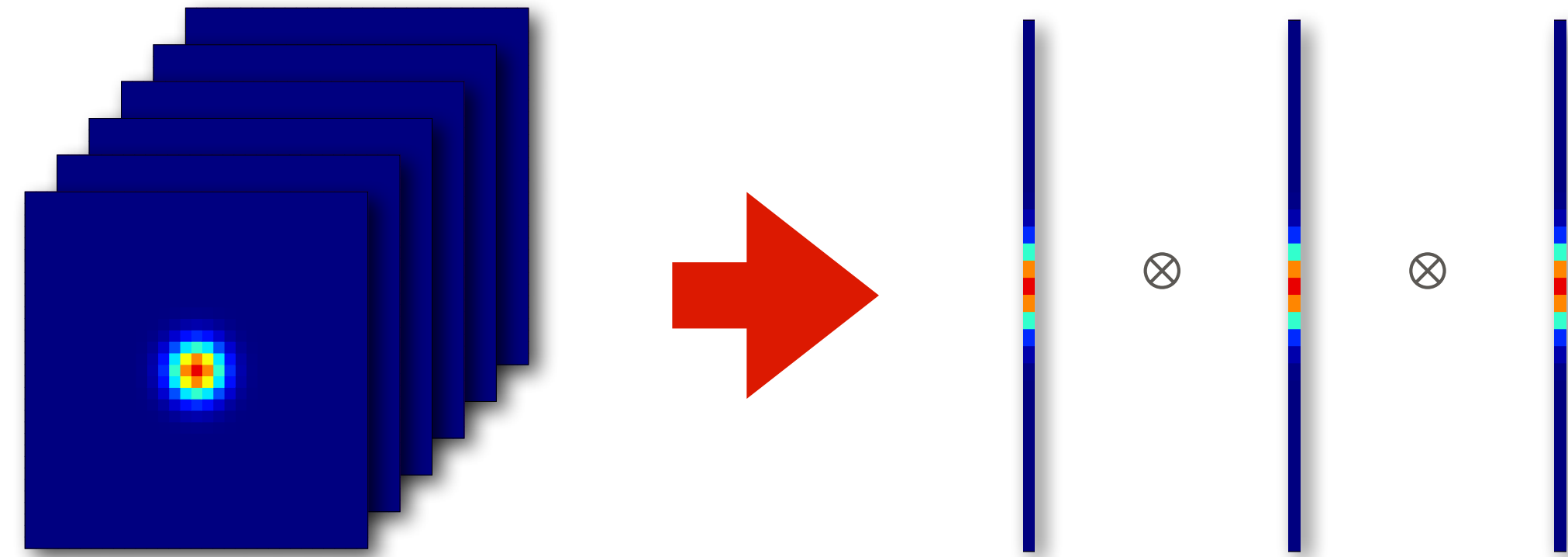
Higher-Order (ND) Tensor: Tensor \mathcal{A}



A simple tensor is given by the outer product of vectors

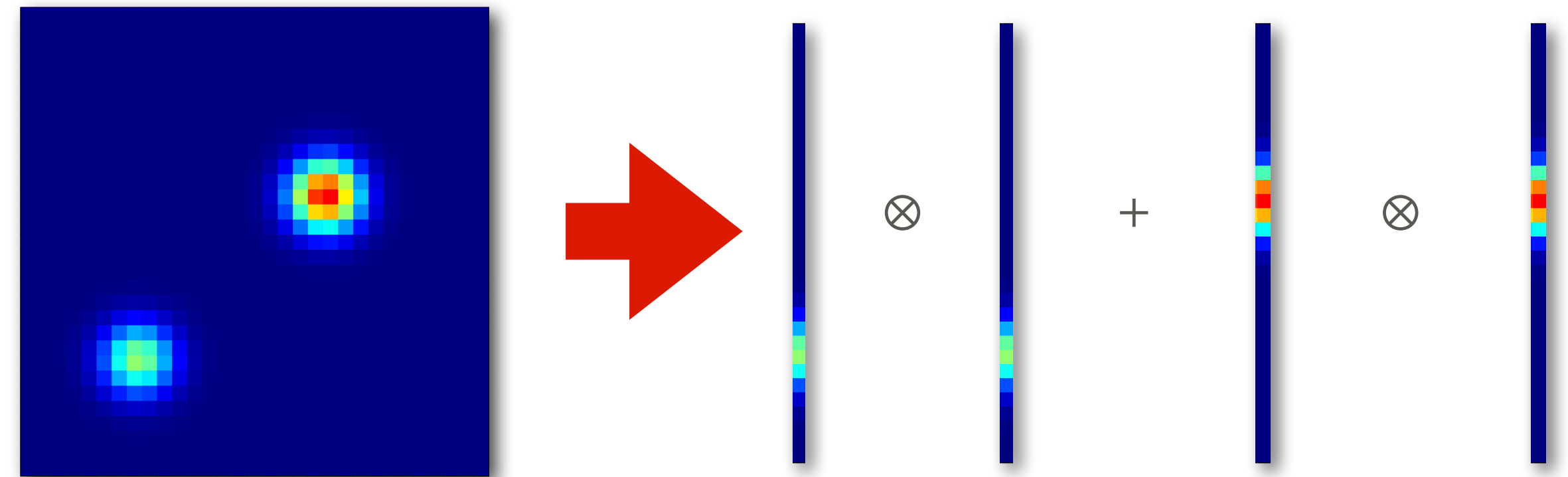
$$\mathcal{A} = \mathbf{a}^{(1)} \otimes \mathbf{a}^{(2)} \otimes \dots \otimes \mathbf{a}^{(D)}$$

$$a_{i_1 i_2 \dots i_D} = a_{i_1}^{(1)} a_{i_2}^{(2)} \dots a_{i_D}^{(D)}$$



and these tensors can be combined to create even more complexity

$$\mathcal{A} = \sum_{k=1}^r \mathbf{a}_k^{(1)} \otimes \mathbf{a}_k^{(2)} \otimes \dots \otimes \mathbf{a}_k^{(D)}$$



In tensor-train terminology, the number of directions is the **dimensionality** and the number of terms in the sum is the **rank**.

Note: Terminology is reversed in the matrix product state community (rank and bond dimension, respectively)!

LOW-RANK TENSOR-TRAIN (TT) REPRESENTATION

The low-rank TT representation reduces the cost of storing the tensor \mathcal{A} by approximating it by lower rank tensor broken up into tensor cores A_j

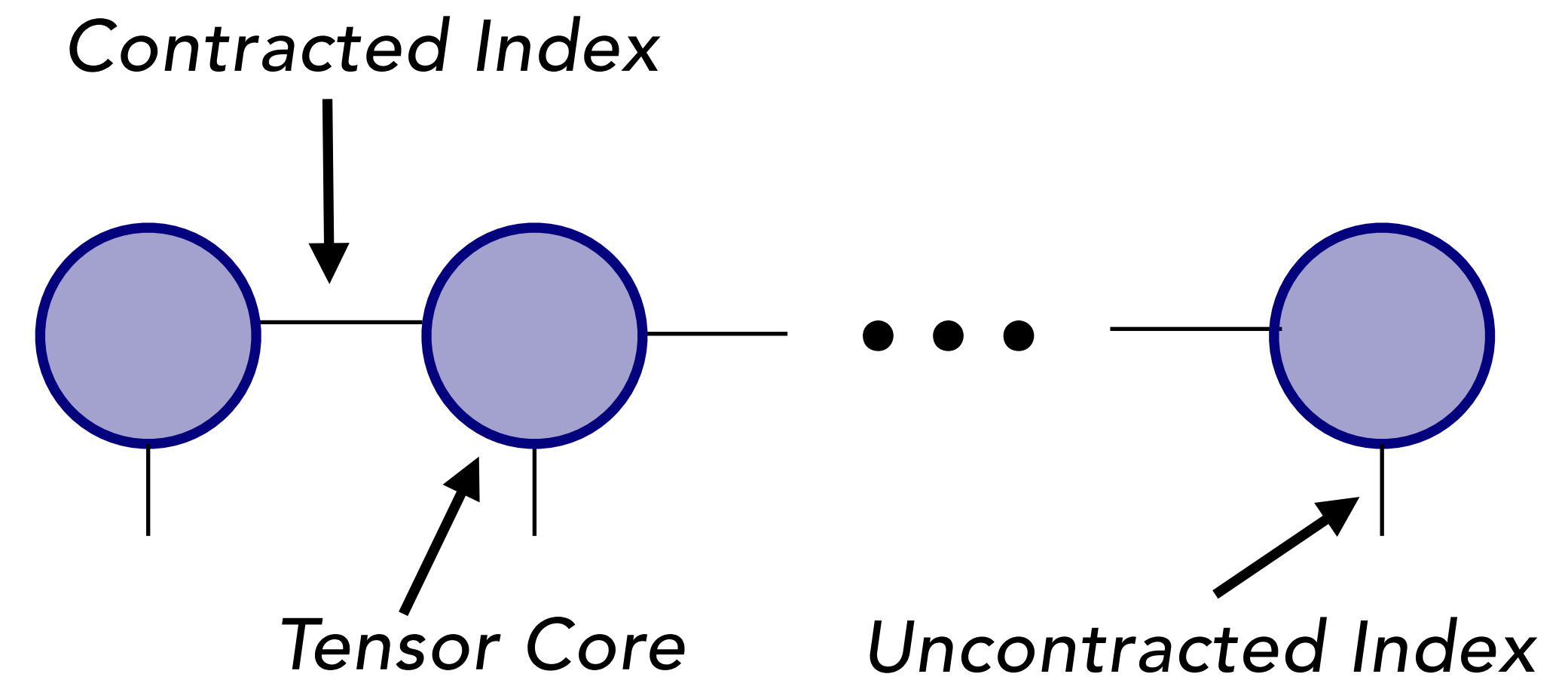
$$\begin{aligned}\mathcal{A}(i_1, \dots, i_d) &\approx \sum_{\alpha_0, \dots, \alpha_d} A_1(\alpha_0, i_1, \alpha_1) A_2(\alpha_1, i_2, \alpha_2) \cdots A_d(\alpha_{d-1}, i_d, \alpha_d) \\ &= \sum_{\alpha_0, \dots, \alpha_d} A_1(i_1, \alpha_1) A_2(\alpha_1, i_2, \alpha_2) \cdots A_d(\alpha_{d-1}, i_d) \\ &= A_1[i_1] A_2[i_2] \cdots A_d[i_d]\end{aligned}$$

α_j : Auxiliary indices for contraction

$A(\alpha_{k-1}, n_k, \alpha_k)$ or $A_j[i_j]$: Tensor core array of size

$r_{k-1} \times n_k \times r_k$ (expressible as a matrix of size $r_{k-1} \times r_k$)

Penrose Notation

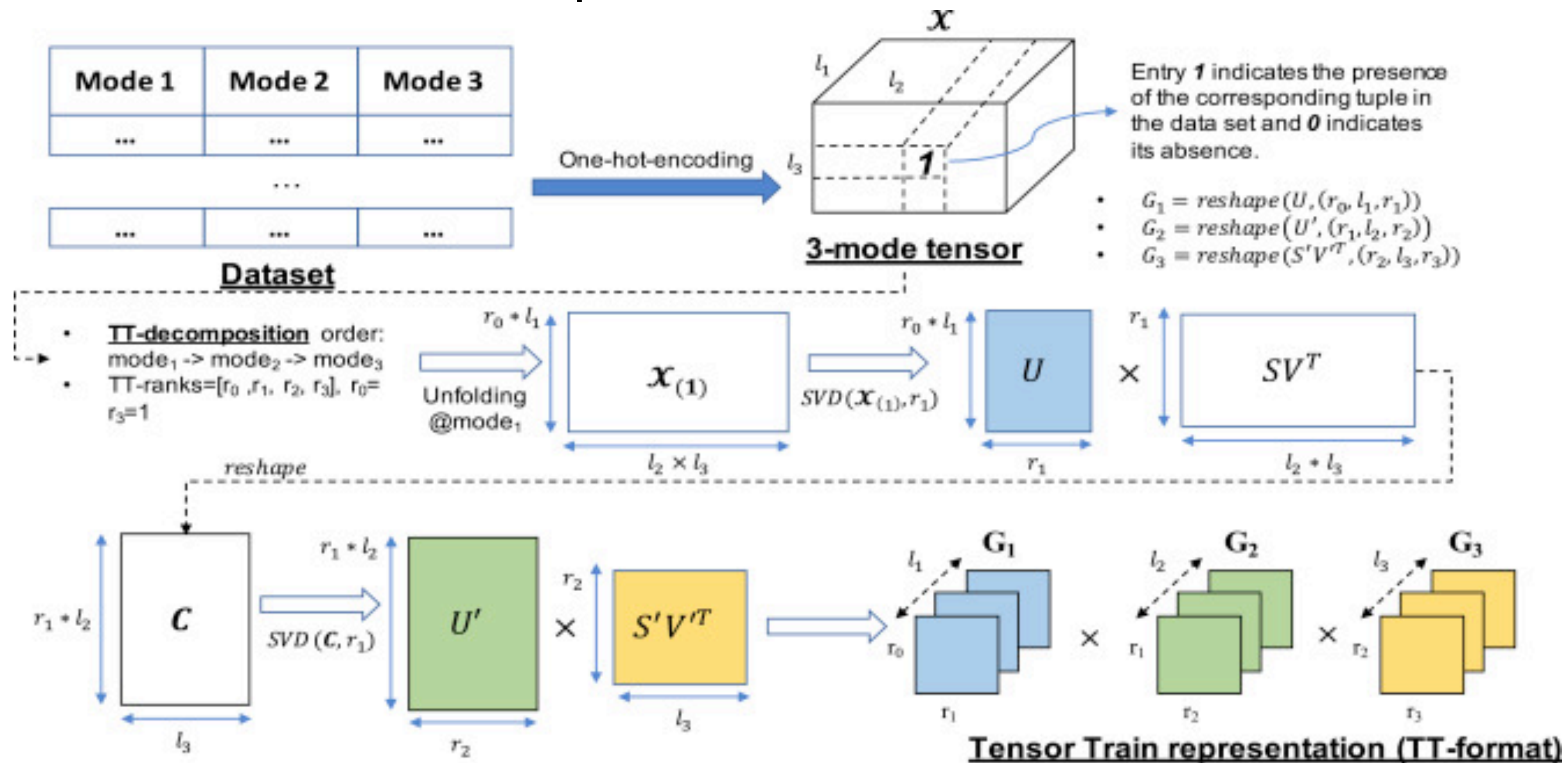


THREE WAYS TO GENERATE TENSOR TRAINS

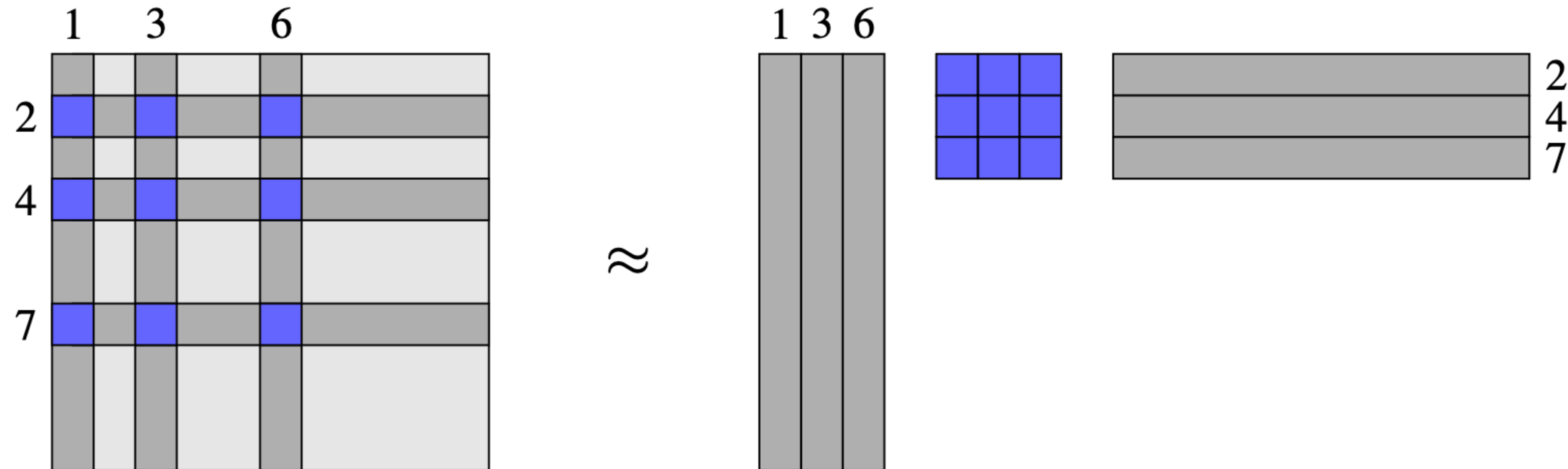
Method One: Generate from known individual cores.

$$\text{Example: } \mathcal{A}(i_1, \dots, i_d) = A_1(\alpha_0, i_1, \alpha_1) A_2(\alpha_1, i_2, \alpha_2) \cdots A_d(\alpha_{d-1}, i_d, \alpha_d)$$

Method Two: Singular Value Decomposition Method



Method Three: Cross Approximation



Built-In Commands in Oseledets TT-Toolbox

Method One and Two: `tensor` `kron`

Method Two: `tensor`

Method Three: `multifuncrs` `multifuncrs2` `amen_cross`

J. Ballani, D. Kressner, "Matrices with Hierarchical Low-Rank Structures," *Exploiting Hidden Structure in Matrix Computations: Algorithms and Applications*, 2173 (2017) 161.

I. Oseledets, E. Tyrtyshnikov, *Linear Algebra Appl.* 432 (2010) 70.

I. Oseledets, *oseledets/TT-Toolbox* (2020) <https://www.github.com/oseledets/TT-Toolbox>.

KEY: Once data is compressed in tensor-train form, computations can be carried out while staying in the data efficient representation, including:

Addition: $E(i_1, i_2, \dots, i_d) = A(i_1, i_2, \dots, i_d) + B(i_1, i_2, \dots, i_d)$

Subtraction: $E(i_1, i_2, \dots, i_d) = A(i_1, i_2, \dots, i_d) - B(i_1, i_2, \dots, i_d)$

Elementwise Multiplication: $E(i_1, i_2, \dots, i_d) = A(i_1, i_2, \dots, i_d)B(i_1, i_2, \dots, i_d)$

Dot Product: $\sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \dots \sum_{i_d=1}^{n_d} A^*(i_1, i_2, \dots, i_d)B(i_1, i_2, \dots, i_d)$

In particular, tensor-train representations are generated with fast adaptive interpolation of multidimensional arrays in TT-Toolbox, and codes are generated to avoid returning to the full grid-based representation.

The tensor-train approach therefore ensures functions are never evaluated everywhere on the original grid, including:

- Functions of TTs (ex. exponential/propagator $e^{-V(\mathbf{x})}$)
- Operators (ex. Heaviside function/projection $\Theta(\mathbf{x} - \bar{\mathbf{x}})$)
- Integrals (ex. Expectation values $\langle x \rangle$)

This points to a central approach to creation of tensor-train codes for molecular simulations:

Translate codes from linear algebra to multilinear algebra, taking care to remain in tensor-train data compressed format throughout.

This is the core principle of the codes we will discuss today: TT-SOFT, TT-Chebyshev, and IPA.



Table of contents



+ Code + Text

Connect



Import Libraries

Hands-On Tutorial on Tensor Trains

What is a tensor train?

Basic Routines

tt_tensor class

tt_matrix class

Advanced functions

1. Linear System Solver

2. Eigenvalue Solver

3. Cross Approximation of a Black-Box Function

3.1. Fast Evaluation

3.2. Integrals

3.3. Tensor Inversion

4. Minimum of a Function

4.1. Example 1: Minimize 4-d Rosenbrock function on a 4-dimensional grid

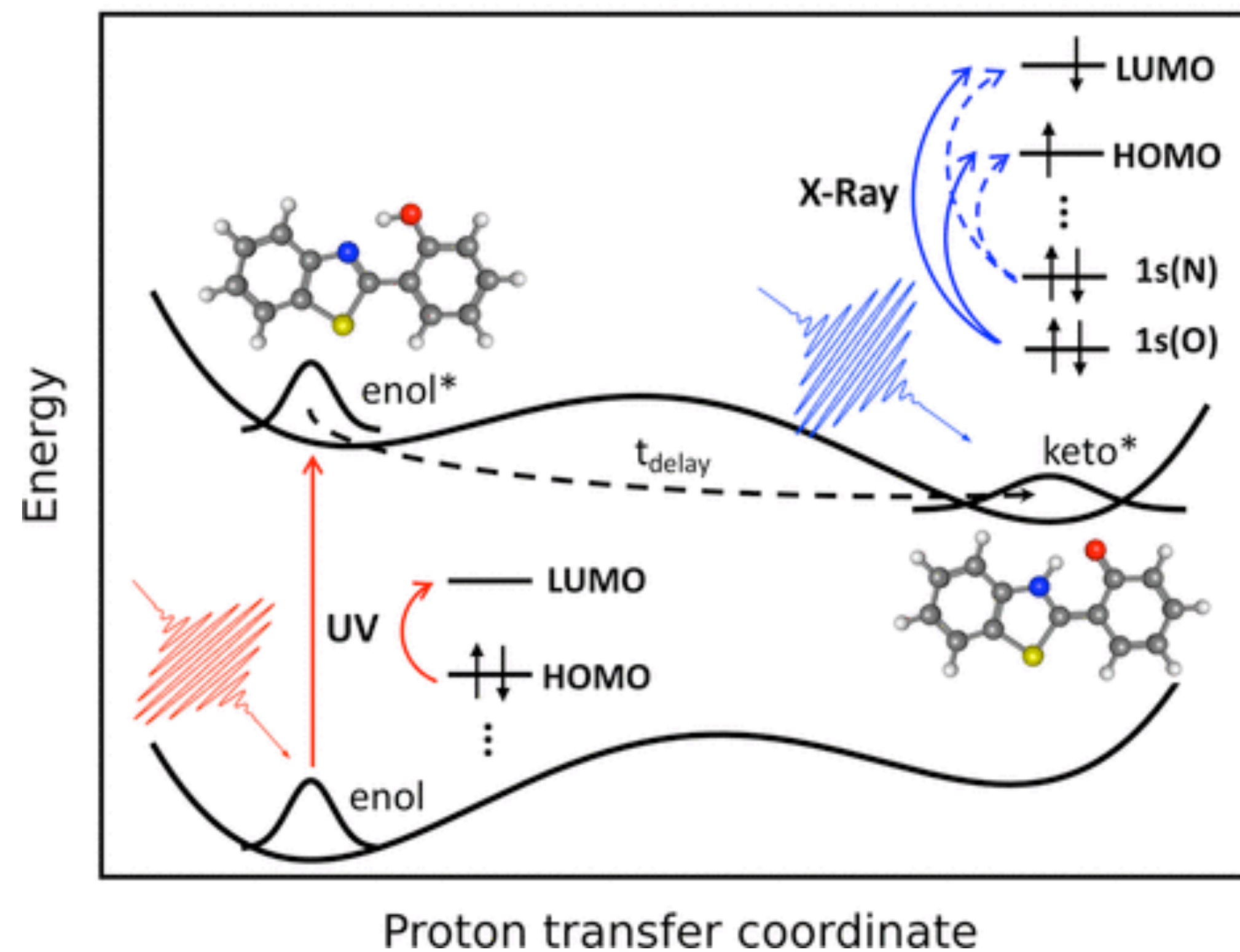
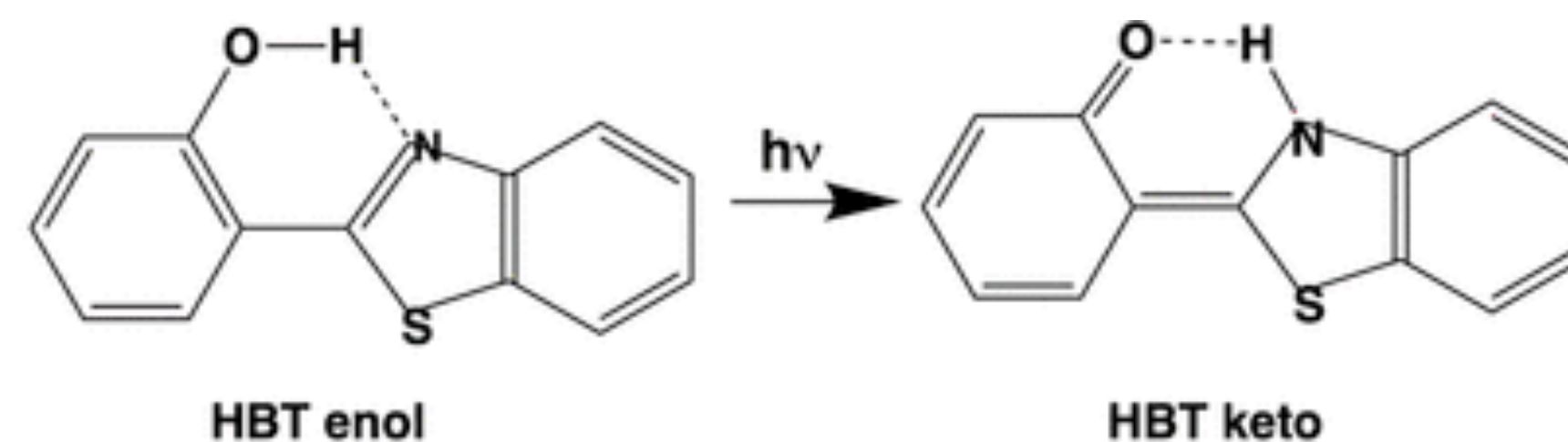
Import Libraries

```
[ ] #!pip install ttpy
#!pip install git+https://github.com/oseledets/ttpy.git@refs/pull/87/head
# As soon as #87 is merged you can drop @... suffix and run.
!pip install git+https://github.com/oseledets/ttpy.git
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple
Collecting git+https://github.com/oseledets/ttpy.git
  Cloning https://github.com/oseledets/ttpy.git to /tmp/pip-req-build-2nh1wddf
  Running command git clone --filter=blob:none --quiet https://github.com/oseledets/ttpy.git /tmp
  Resolved https://github.com/oseledets/ttpy.git to commit a50d5e0ce2a033a4b1aa703715cb85d715b9b3
  Running command git submodule update --init --recursive -q
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Building wheels for collected packages: ttpy
  Building wheel for ttpy (pyproject.toml) ... done
  Created wheel for ttpy: filename=ttpy-1.2.0-cp310-cp310-linux_x86_64.whl size=3546003 sha256=dd
  Stored in directory: /tmp/pip-ephem-wheel-cache-605g0su8/wheels/0f/9c/16/16342a640cb36d2dad60b1
Successfully built ttpy
Installing collected packages: ttpy
Successfully installed ttpy-1.2.0
```

**TOPIC 1: EXACT QUANTUM DYNAMICS IN
HIGH DIMENSIONALITY WITH TT-SOFT**

TT-SOFT FOR SIMULATION OF QUANTUM EFFECTS IN COMPLEX CHEMICAL SYSTEMS



BASIS OF THE METHOD: SPLIT-OPERATOR FOURIER TRANSFORM (SOFT) QUANTUM DYNAMICS

Consider the wavefunction $\psi(t)$ in the basis of n equidistant delta functions $\delta(x - x_k)$ in the range $\{x_{\min}, x_{\max}\}$.

The Suzuki-Trotter expansion approximates the propagator as

$$e^{-i\hat{H}\tau} = e^{-i\hat{V}\tau/2} e^{-i\hat{p}^2\tau/(2m)} e^{-i\hat{H}\tau/2}$$

such that the wavefunction is propagated for a short time τ as

$$\psi(t + \tau) = e^{-i\hat{V}\tau/2} \int dp e^{ixp} e^{-i\hat{p}^2\tau/(2m)} \frac{1}{2\pi} \int dx' e^{-ipx'} e^{-i\hat{V}\tau/2} \psi(t).$$

Problem: Grid-based implementation rapidly reaches computational memory limits.

M. D. Feit, J. A. Fleck, A. Steiger, J. Comput. Phys. 47 (1982) 412.

M. Soley, A. Markmann, V. S. Batista, J. Phys. Chem. B, 119 (2015) 715.

ADAPTATION TO TENSOR-TRAIN FORMAT: INITIALIZATION

Initialize the wavepacket as a rank-one tensor train

$$\psi(\mathbf{x}; t_0) = \prod_{j=1}^d \psi_j(x_j; t_0)$$

$$\psi(x_1, \dots, x_d; t) = \sum_{\alpha_0, \dots, \alpha_d} \psi_1(x_1, \alpha_1; t) \psi_2(\alpha_1, x_2, \alpha_2; t) \cdots \psi_d(\alpha_{d-1}, x_d; t)$$

Represent potential as tensor train by construction/SVD/cross approximation.

$$V(x_1, \dots, x_d; t) = \sum_{\alpha_0, \dots, \alpha_d} V_1(x_1, \alpha_1; t) V_2(\alpha_1, x_2, \alpha_2; t) \cdots V_d(\alpha_{d-1}, x_d; t)$$

[Micheline B. Soley,* P. E. Videla,* E. T. J. Nibbering, V. S. Batista, \(2022\) J. Phys. Chem. Lett., 18 \(2022\) 8254.](#)

S. M. Greene, V. S. Batista, JCTC 13 (2017) 4034.

ADAPTATION TO TENSOR-TRAIN FORMAT: PROPAGATOR

Evaluate the potential propagator as truncated power series by construction:

$$e^{-iV\tau/\hbar} = \sum_{n=0}^N \frac{(-iV\tau/\hbar)^n}{n!}$$

Equivalently, the exponential may be generated via the scaling-and-squaring method, cross approximation, etc.

Evaluate the kinetic propagator as a rank-one tensor train:

$$e^{-i\mathbf{p}\cdot\mathbf{m}^{-1}\cdot\mathbf{p}\tau/(2\hbar)} = \prod_{j=1}^d e^{-ip_j^2\tau/2m_j\hbar}$$

ADAPTATION TO TENSOR-TRAIN FORMAT: PROPAGATION

Calculate required the FFT/IFFT (built-in):

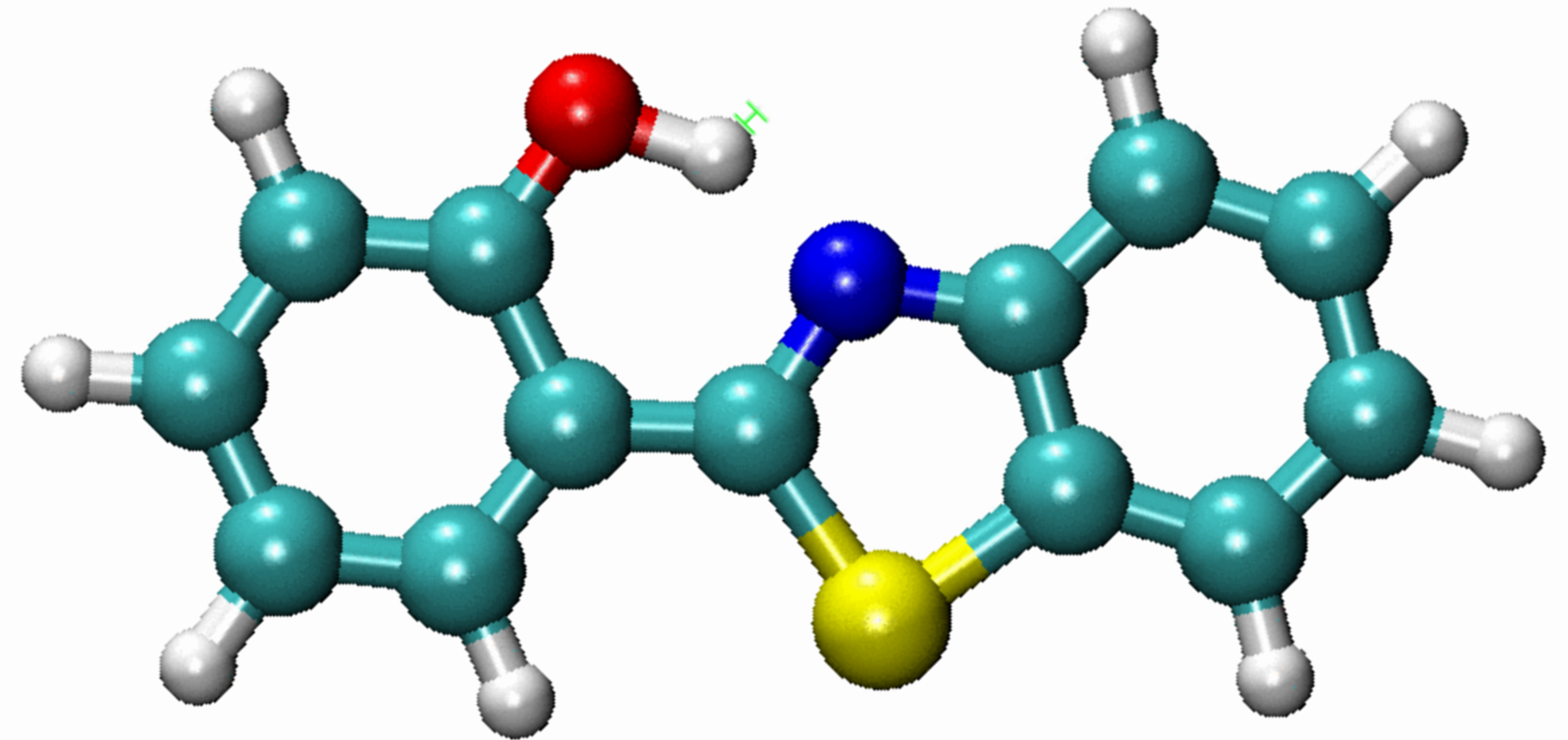
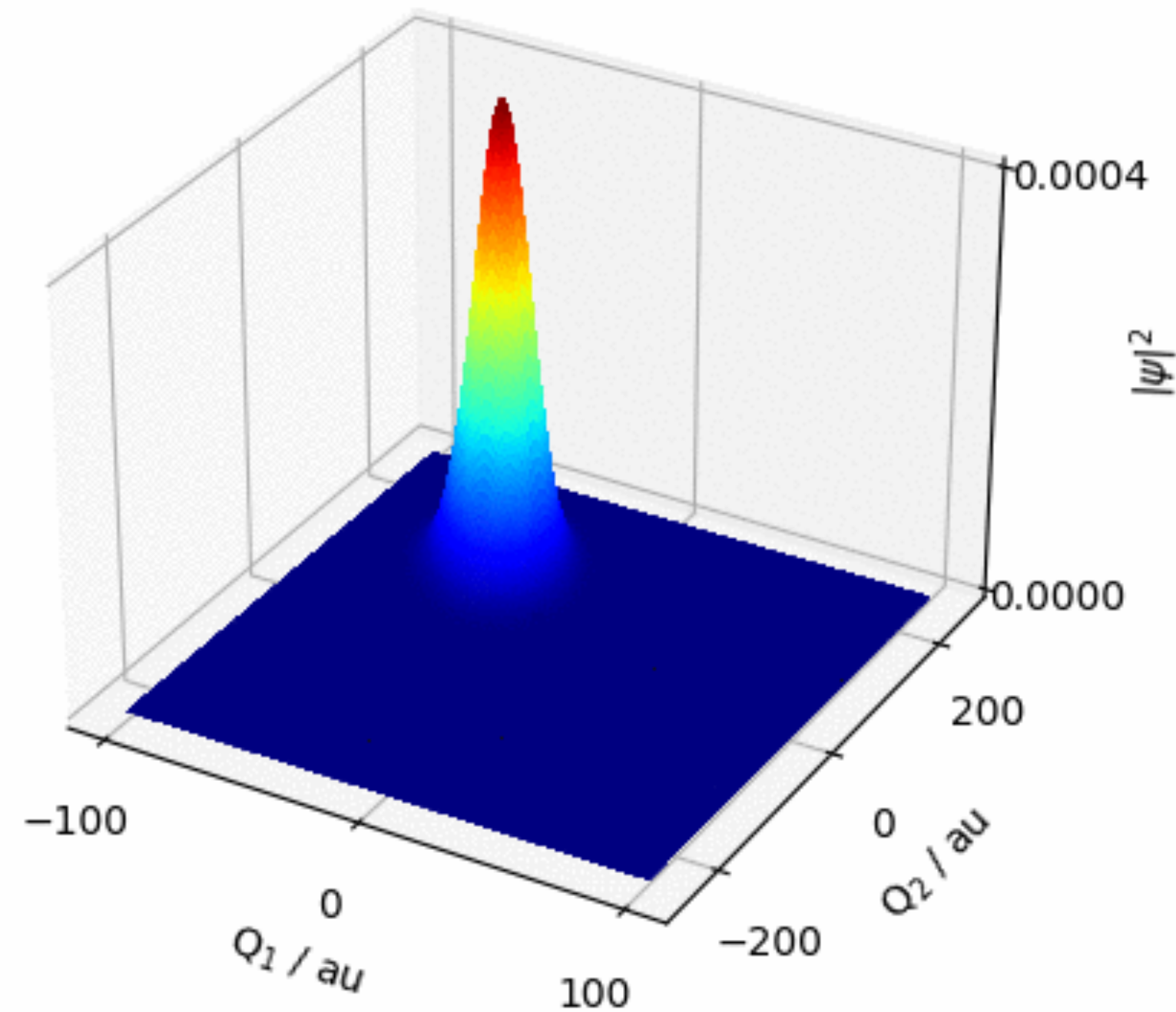
$$\tilde{\psi}(p_1, p_2, \dots, p_d; t) = \sum_{\alpha_1, \alpha_2, \dots, \alpha_{d-1}} (2\pi\hbar)^{-d/2} \int d\mathbf{x} e^{-i\mathbf{x}\cdot\mathbf{p}/\hbar} \psi_1(x_1, \alpha_1; t) \psi_2(\alpha_1, x_2, \alpha_2; t) \cdots \psi_d(\alpha_{d-1}, x_d; t)$$

Since the algebraic manipulations in TT-SOFT increase the wavefunction rank during the simulation, the tensor-train rank is rounded after each propagation step.

TT-SOFT RESULTS: QUANTUM DYNAMICS

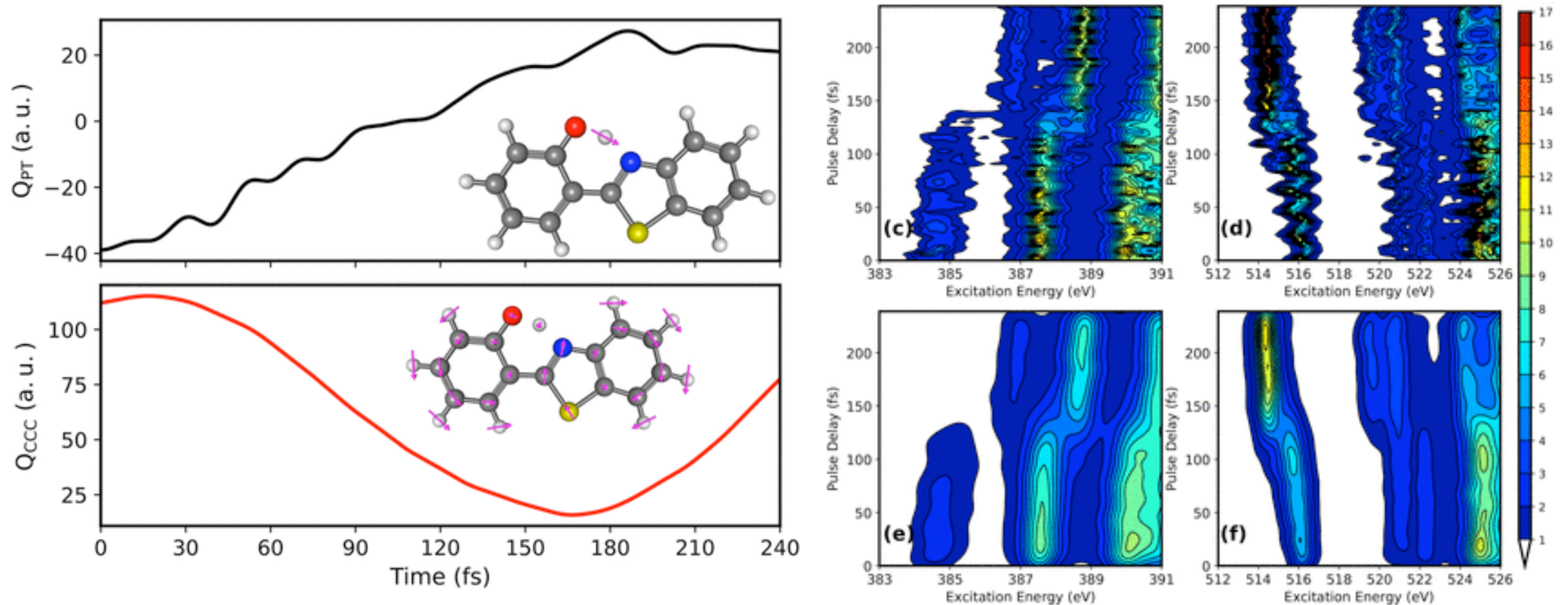
TT-SOFT propagation in an ab initio potential: <https://github.com/michelinesoley/HBT>

Time = 0.00 fs



TT-SOFT successfully computes dynamics of a molecular system with full inclusion of quantum effects in 69D.

TT-SOFT RESULTS: PUMP-PROBE SPECTRA



The method thereby enables exact simulation of molecular phenomena beyond reach with standard exact grid-based methods.



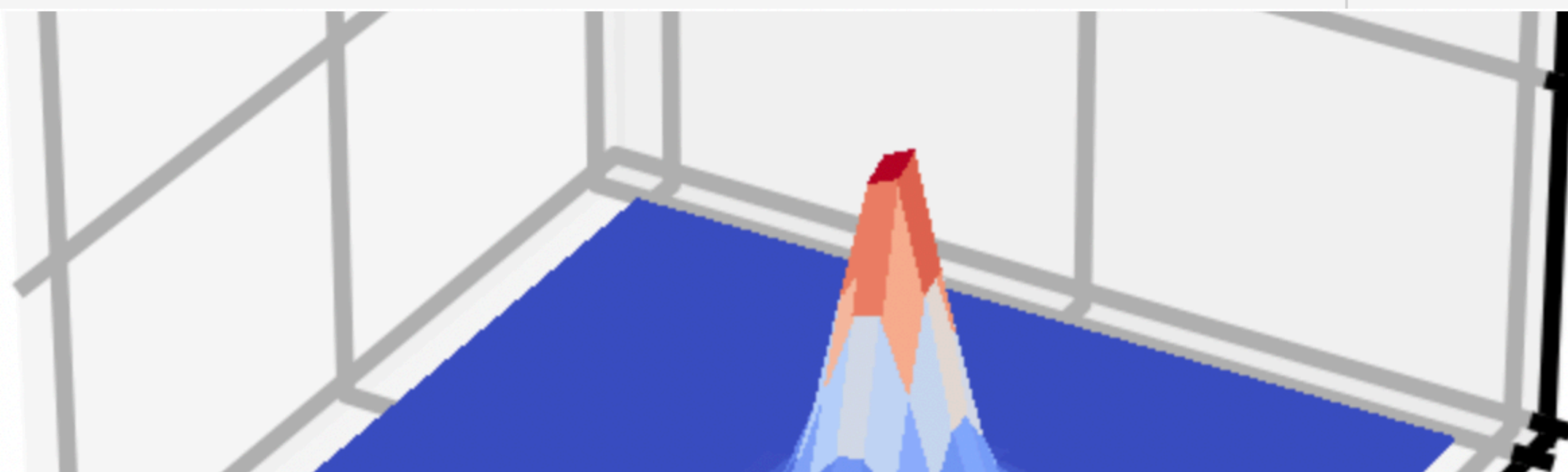
+ Code + Text

Connect ▾



```
[ ]
    #print('r0=',tt_psi[0].r)
# plot wavepacket components
#if js%1==0:
    if wfflag == 1: #check param to plot
        ttps1=tt_psi[0].full()
        ttps1=np.reshape(ttps1,[nx[0]]*dim)
        plt.figure(dpi=600)
        ax= plt.subplot(3,2,2, projection='3d')
        if qmodes == 0:
            ax.plot_surface(x, y, np.abs(ttps1[:, :, ]), cmap=cm.coolwarm, antialiased=False)
        if qmodes == 1:
            ax.plot_surface(x, y, np.abs(ttps2[:, :, nsl]), cmap=cm.coolwarm, antialiased=False)
        ax.set_zlim3d(0,1)

        plt.pause(.02)
        if js < nsc-1:
            plt.clf()
```

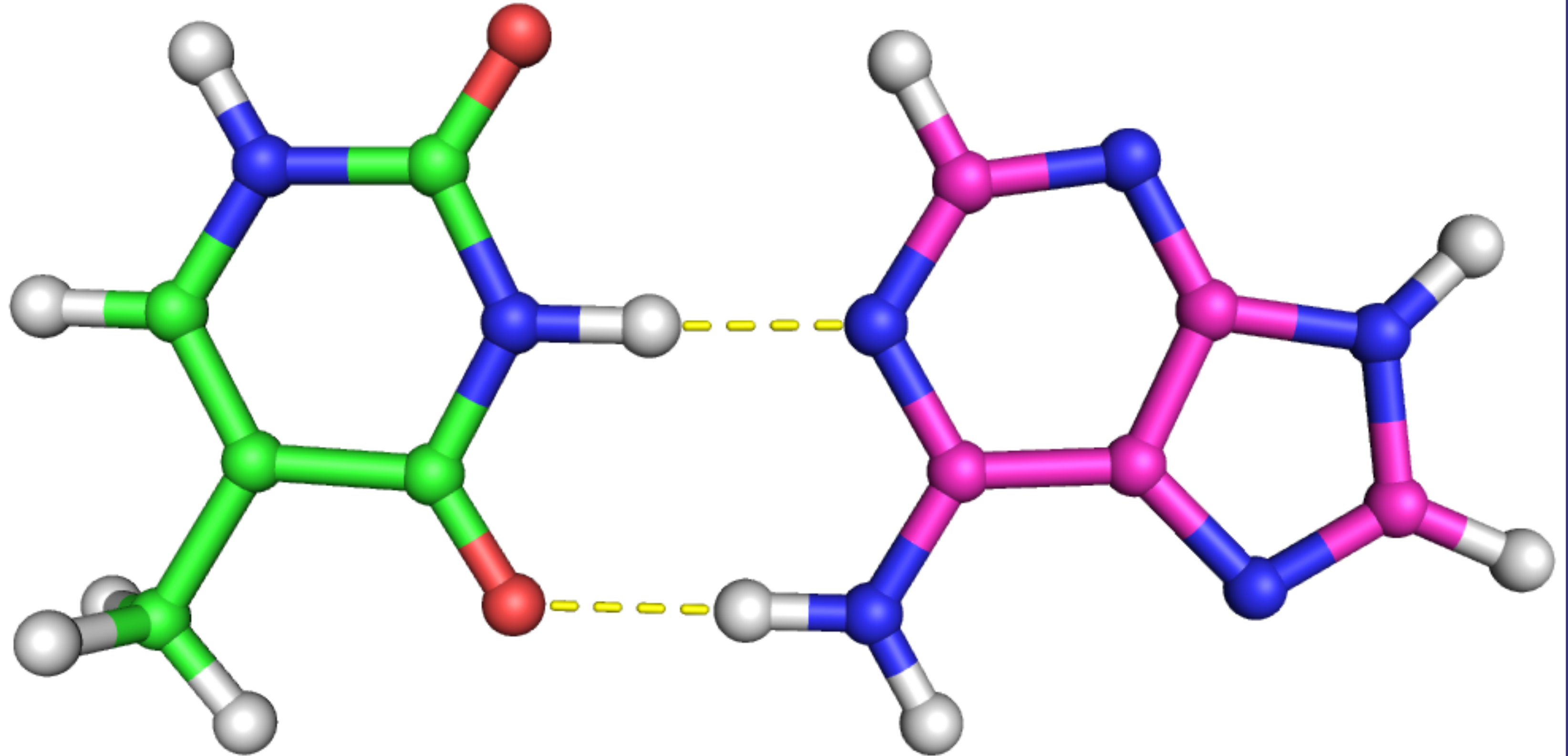
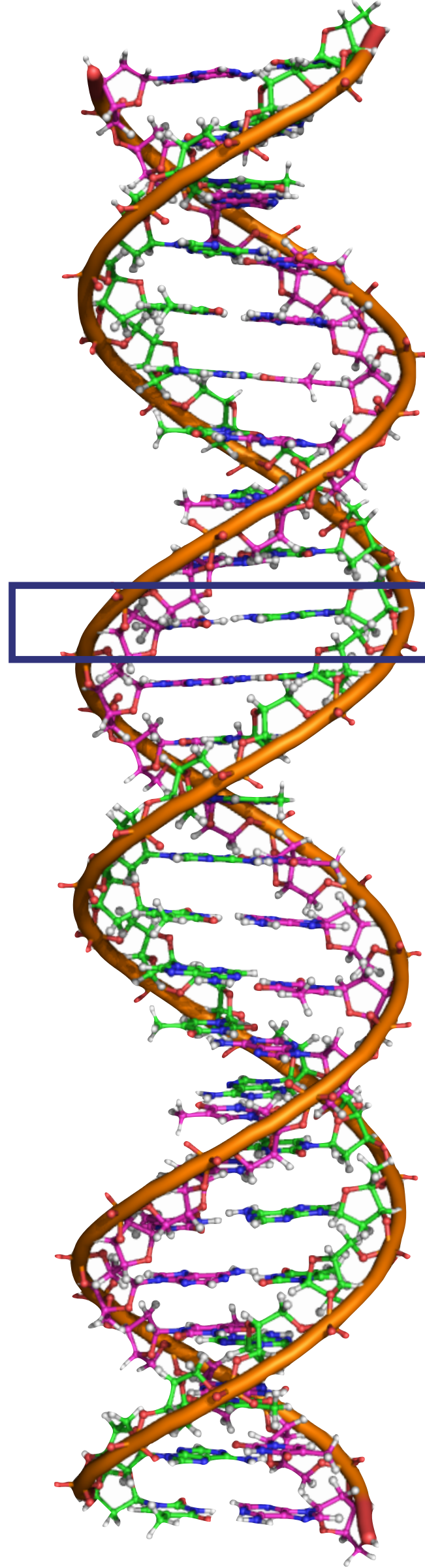


0.5

0 0

**EXACT QUANTUM DYNAMICS IN HIGH
DIMENSIONALITY: TT-CHEBYSHEV**

ACCURATE QUANTUM DYNAMICS IN HIGH DIMENSIONALITY



Micheline B. Soley, P. Bergold, A. A. Gorodetsky, V. S. Batista, *JCTC*, 18 (2022) 25.

N. Lyu*, E. Mulvihill*, **Micheline B. Soley**, E. Geva, V. S. Batista, *JCTC*, 19 (2023) 1111.

Micheline B. Soley*, P. E. Videla*, E. T. J. Nibbering, V. S. Batista, *J. Phys. Chem. Lett.*, 13 (2022) 8354.

N. Lyu, **Micheline B. Soley**, V. S. Batista, *JCTC*, 18 (2022) 3327.

Micheline B. Soley, P. Bergold, V. S. Batista, *JCTC* 17 (2021) 3280.

Chebyshev Polynomials

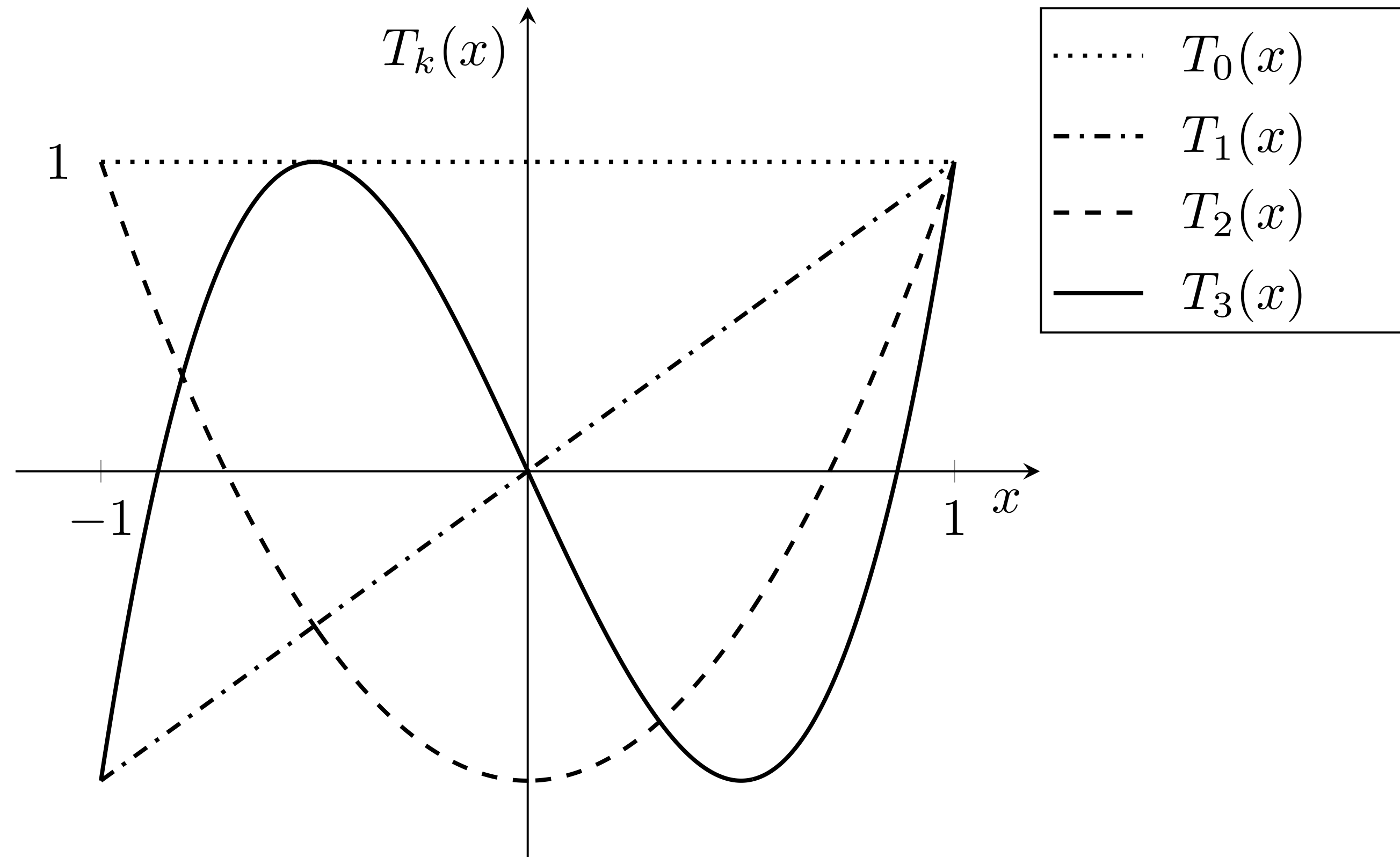
For all integers $k \geq 0$ and all $x \in [-1, 1]$, the k th Chebyshev polynomial is defined as

$$T_k(x) = \cos(k \arccos(x))$$

Recurrence relation:

$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x)$$

$$T_0(x) = 1, T_1(x) = x$$



[Micheline B. Soley, P. Bergold, A. A. Gorodetsky, V. S. Batista, JCTC, 18 \(2022\) 25.](#)

L. N. Trefethen, *Approximation Theory and Approximation Practice*, SIAM: Philadelphia, 2013.

H. Tal-Ezer, R. Kosloff, *J. Chem. Phys.* 81 (1984) 3967.

Chebyshev Expansion of Complex-valued Functions

Chebyshev polynomials can be used to approximate a given complex-valued function f via its Fourier series representation

$$g(x) = f(\cos(x))$$
$$g(x) = \sum_{k=0}^{\infty} (2 - \delta_{k,0}) a_k \cos(kx), \quad a_k = \frac{1}{\pi} \int_0^{\pi} g(x) \cos(kx) dx$$

such that $f(y) = g(\arccos(y))$ can be represented in terms of Chebyshev polynomials for $y \in [-1, 1]$

$$f(y) = \sum_{k=0}^{\infty} (2 - \delta_{k,0}) c_k T_k(y), \quad c_k = \frac{1}{\pi} \int_{-1}^1 \frac{dy}{\sqrt{1-y^2}} f(y) T_k(y)$$

Chebyshev Approximant of Complex-valued Functions

This Chebyshev expansion of f can then be used to approximate f as the linear combination of the first N Chebyshev polynomials

$$f(y) \approx S_N f(y) = \sum_{k=0}^{N-1} (2 - \delta_{k,0}) c_k T_k(y)$$

The resulting Chebyshev approximant $S_N f$ is a polynomial of degree N that is known to be close to the polynomial of the same degree with minimal error in the interval $[-1, 1]$.

Chebyshev Propagation in Discrete Representations

We obtain an approximation of the propagator as applied to the wavefunction according to the Chebyshev expansion of complex-valued functions:

$$\Psi(t) = e^{-itH}\Psi(0)$$

$$e^{itH} \approx e^{-it^+} \sum_{k=0}^{N-1} (2 - \delta_{k,0}) (-i)^k J_k(t^-) T_k(H_0)$$

$$t^\pm = \frac{t}{2} (b \pm a)$$

$$H_0 = \frac{2}{b-a} \left(H - \frac{b+a}{2} I_D \right)$$

Fast convergence is typically obtained for e^{-ity} since it is a smooth function, with error falling as the N^{th} order in $|t^-|/(2N)$ for sufficiently large N .

DISCRETE TENSOR-TRAIN IMPLEMENTATION: HAMILTONIAN

Discrete low-rank TT representations are generated for the wavefunction \mathcal{W} and potential energy operator \mathcal{V} , and the action of the kinetic energy operator on the wavefunction $\hat{\mathcal{T}}\mathcal{W}$.

Here, $\hat{\mathcal{T}}\mathcal{W}$ is found in terms of the Laplacian (to take advantage of highly-efficient implementations of multidimensional discrete Fourier transforms of tensor trains to switch between position and momentum space) to generate the Hamiltonian $\hat{\mathcal{H}}$.

The discrete Hamiltonian is then rescaled as in standard Chebyshev propagation

$$\hat{\mathcal{H}}_0 = \frac{2}{E_{\max} - E_{\min}} \left(\hat{\mathcal{H}} - \frac{E_{\max} + E_{\min}}{2} \hat{\mathcal{I}} \right)$$

where $\hat{\mathcal{I}}$ is the identity on the tensor space.

DISCRETE TENSOR-TRAIN IMPLEMENTATION: PROPAGATION

The propagated wavefunction $\Psi(t)$ is then approximated with N Chebyshev polynomials

$$\Psi(t) = e^{-it\hat{H}}\Psi(0) \approx e^{-it^+} \sum_{k=0}^{N-1} (2 - \delta_{k,0})(-i)^k J_k(t^-) T_k(\hat{\mathcal{H}}_0) \mathcal{W}_0$$

where we employ the Chebyshev Clenshaw algorithm or the recurrence relation

$$\begin{aligned} T_0(\hat{\mathcal{H}}_0) \mathcal{W}_0 &= \mathcal{W}_0 \\ T_1(\hat{\mathcal{H}}_0) \mathcal{W}_0 &= \hat{\mathcal{H}}_0 \mathcal{W}_0 \\ T_{k+1}(\hat{\mathcal{H}}_0) \mathcal{W}_0 &= 2\hat{\mathcal{H}}_0 T_k(\hat{\mathcal{H}}_0) \mathcal{W}_0 - T_{k-1}(\hat{\mathcal{H}}_0) \mathcal{W}_0, \quad \text{for } k \geq 1 \end{aligned}$$

The same Chebyshev propagation scheme can be readily implemented using the continuous analogue **functional tensor-train decomposition**.

CONTINUOUS ANALOGUE OF THE TENSOR-TRAIN DECOMPOSITION

In place of discrete numerical entries, functional tensor trains are composed of univariate functions $f_k^{(ij)}$.

$$f(x_1, x_2, \dots, x_d) = \sum_{i_0=1}^{r_0} \sum_{i_1=1}^{r_1} \cdots \sum_{i_d=1}^{r_d} f_1^{(i_0 i_1)}(x_1) f_2^{(i_1 i_2)}(x_2) \cdots f_d^{(i_{d-1} i_d)}(x_d)$$

$$f(x_1, x_2, \dots, x_d) = \mathcal{F}_1(x_1) \mathcal{F}_2(x_2) \cdots \mathcal{F}_d(x_d)$$

$$\mathcal{F}_k = \begin{bmatrix} f_k^{(11)}(x_k) \cdots f_k^{(1r_k)}(x_k) \\ \vdots \quad \ddots \quad \vdots \\ f_k^{(r_{k-1},1)}(x_k) \cdots f_k^{(r_{k-1}r_k)}(x_k) \end{bmatrix}$$

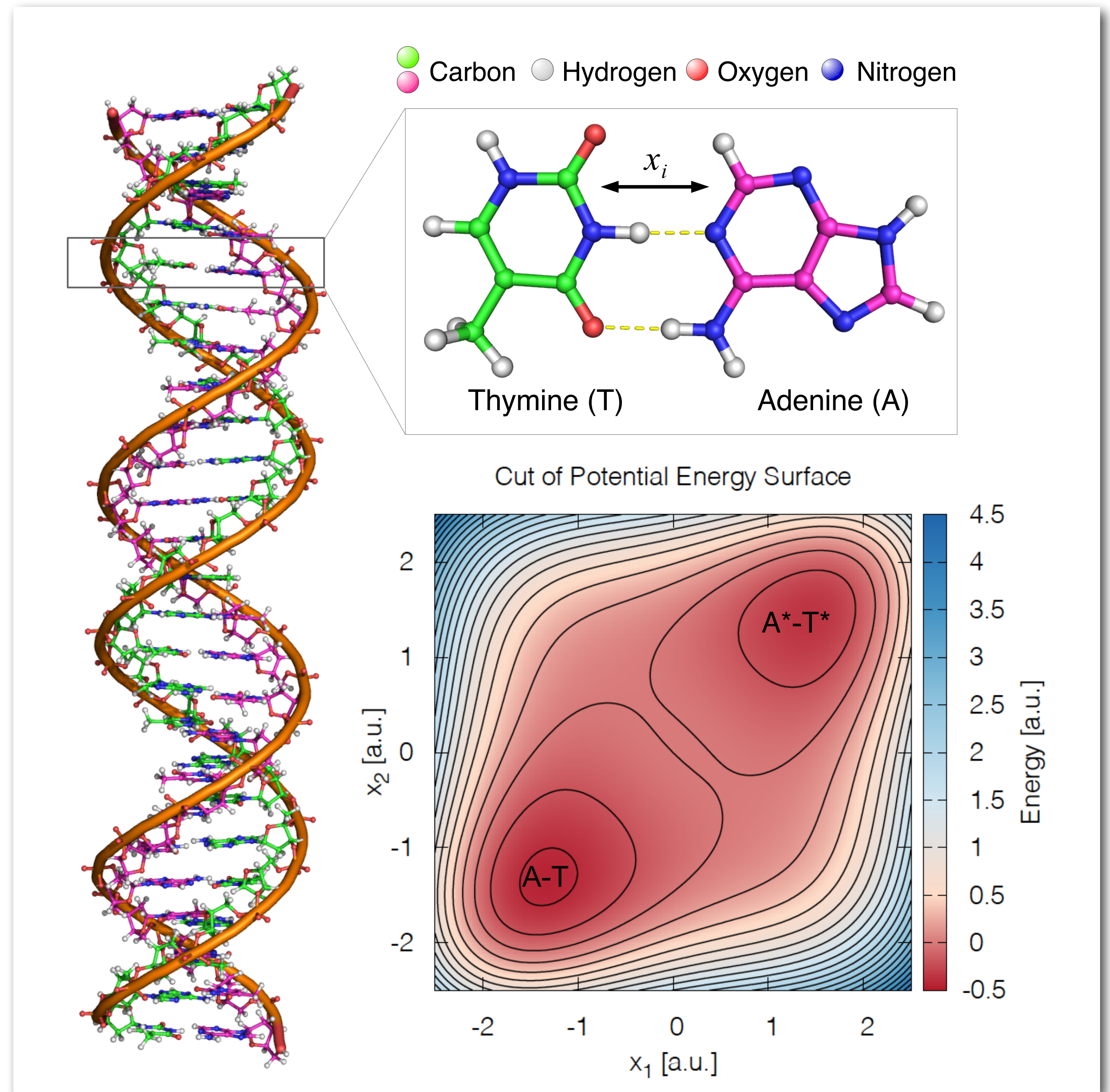
Result: Efficient gradients, integrals, and correlation functions in high dimensionality with likewise ease of calculations

[Micheline B. Soley, P. Bergold, A. A. Gorodetsky, V. S. Batista, JCTC, 18 \(2022\) 25.](#)

[A. Gorodetsky, J. D. Jakeman, J. Comput. Phys. 374 \(2018\) 1219.](#)

[A. Gorodetsky, A. A. Compressed Continuous Computation \(C3\) Library. https://github.com/goroda/Compressed-Continuous-Computation.](https://github.com/goroda/Compressed-Continuous-Computation)

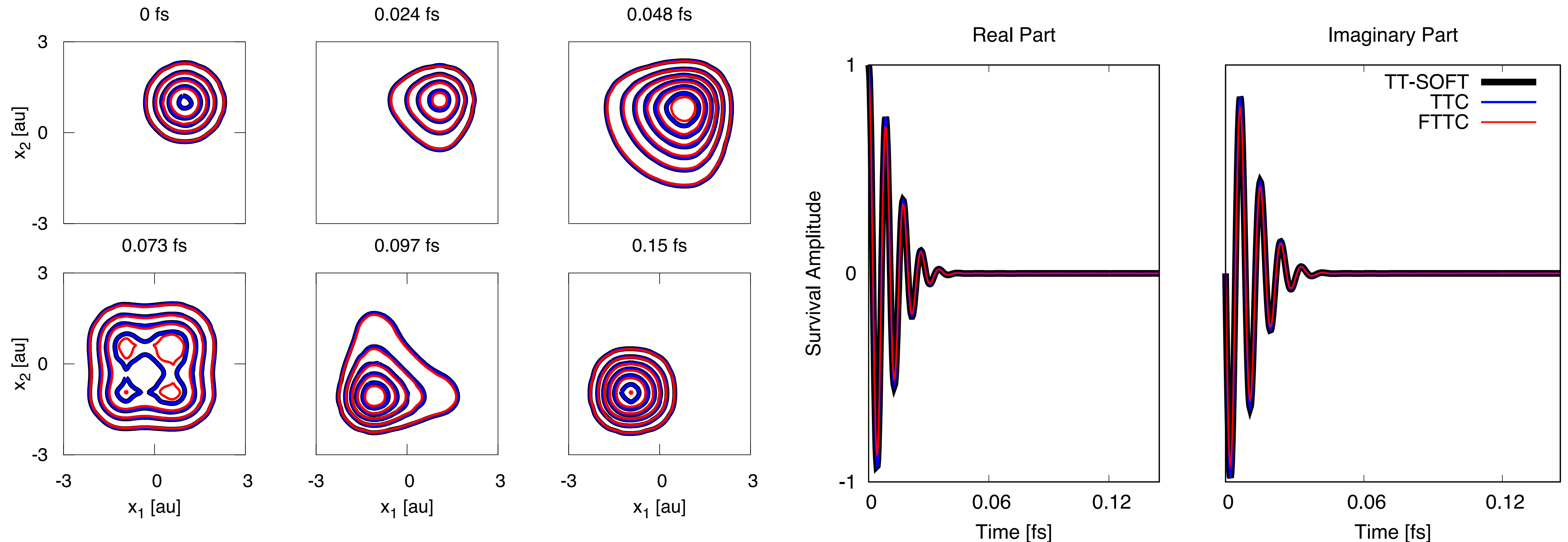
FUNCTIONAL TENSOR TRAIN CHEBYSHEV (FTTC) DYNAMICS RESULTS



APPLICATION: HYDROGEN BONDING IN DNA

Probability Density Dynamics, Uncoupled Bath

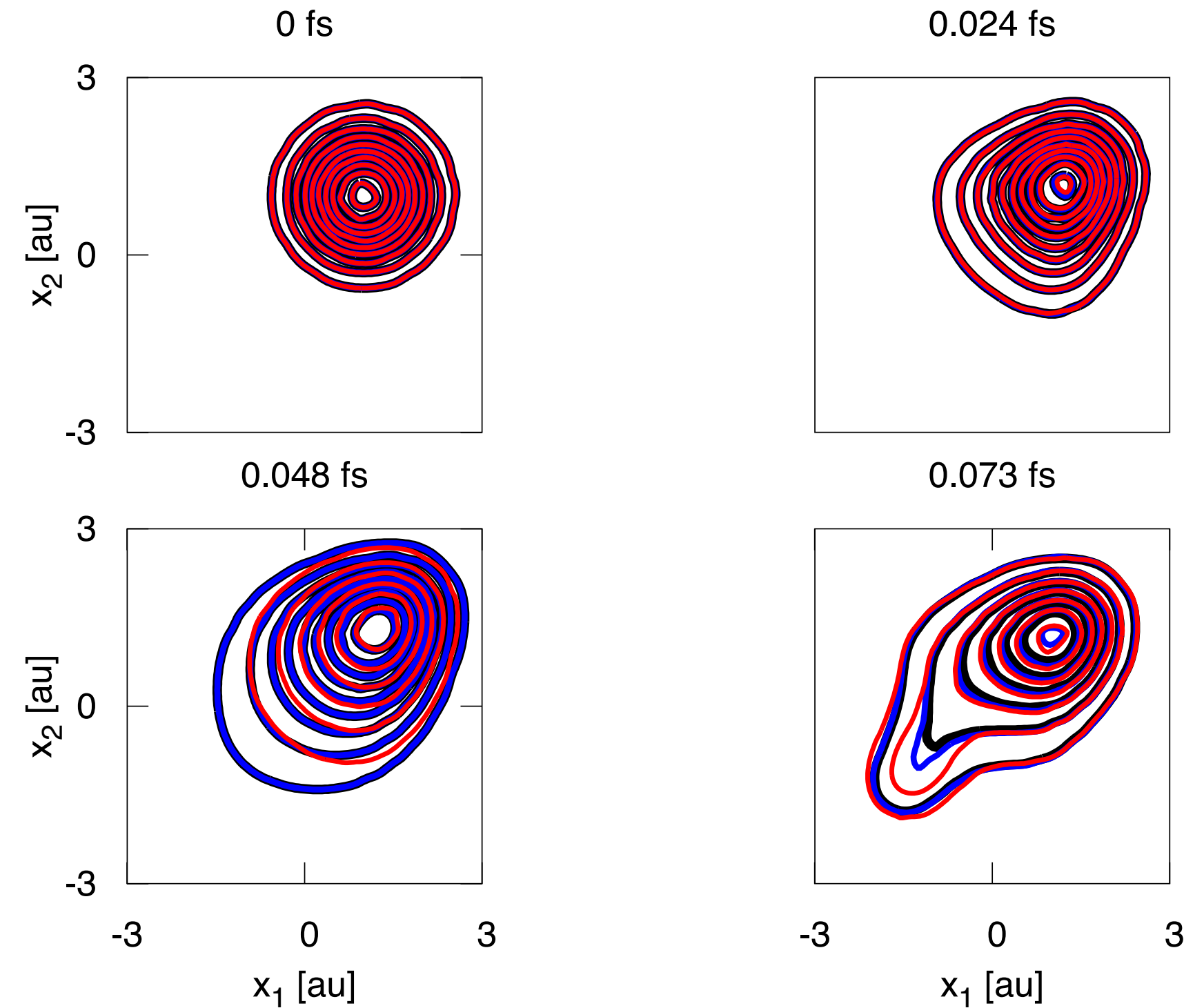
Survival Amplitude



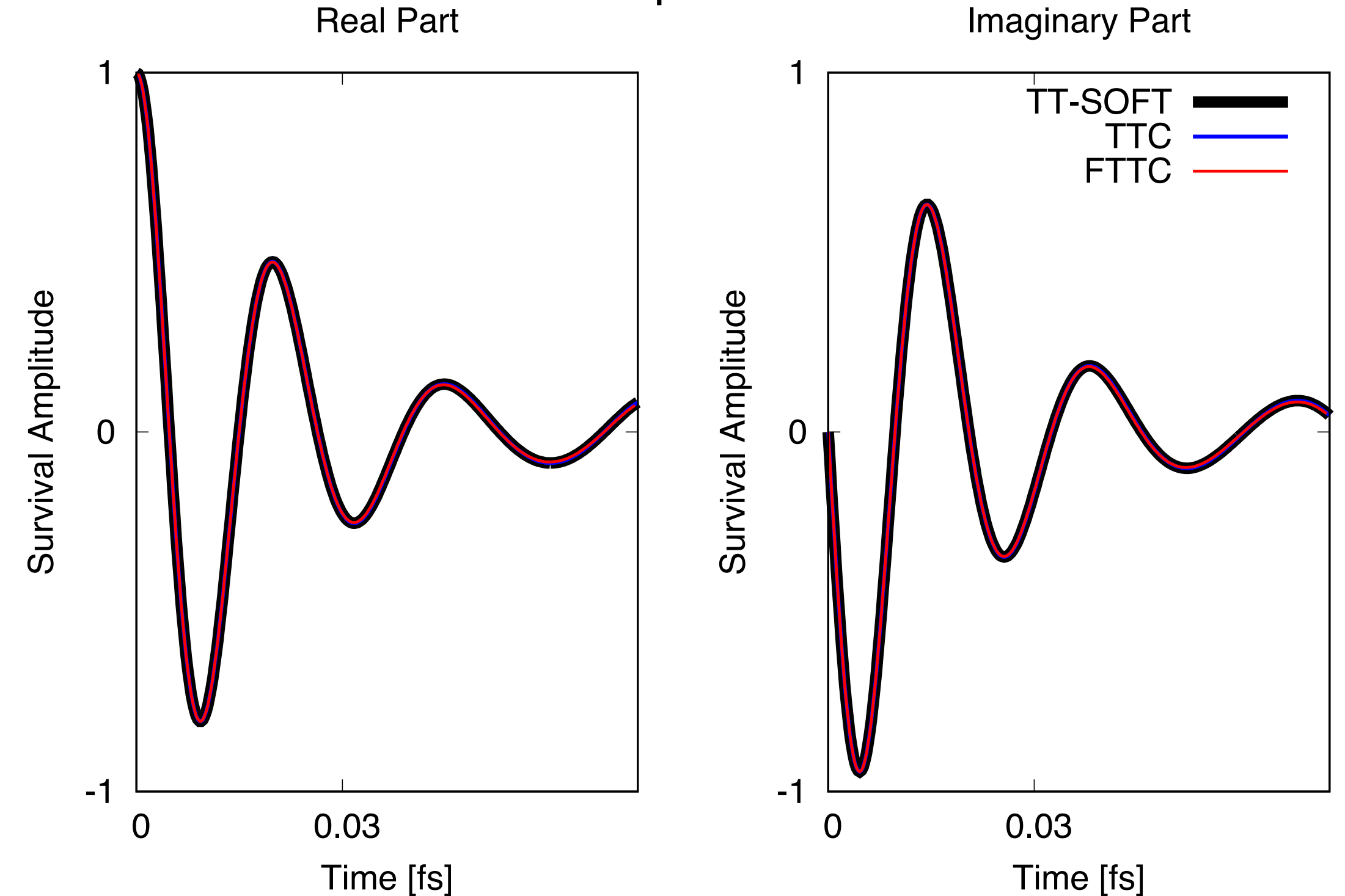
FTTC extends the Chebyshev method from simulation of four-atom systems to molecular systems in ***50 dimensions.***

APPLICATION: HYDROGEN BONDING IN DNA

Probability Density Dynamics, Anharmonically Coupled Bath



Survival Amplitude



**FTTC successfully determines molecular dynamics
even with significant coupling of atomic motion between modes.**

IMPACT OF FTTC

The success of the functional tensor-train decomposition should find wide applicability in studies requiring computations of gradients, integrals, and correlation functions of systems with high dimensionality.

And, the success of FTTC invites its use for other not only for simulations of quantum reaction dynamics in general, but also as a general method to obtain numerical solutions of linear systems in high dimensionality.



Table of contents

Python Functional Tensor-Train Chebyshev (FTTC) Dynamics for Simulation of Hydrogen Bonding in a Highly-Multidimensional DNA Chain

Compressed Continuous Computation Python (c3py) Library.

FTTC Propagation

Laplace

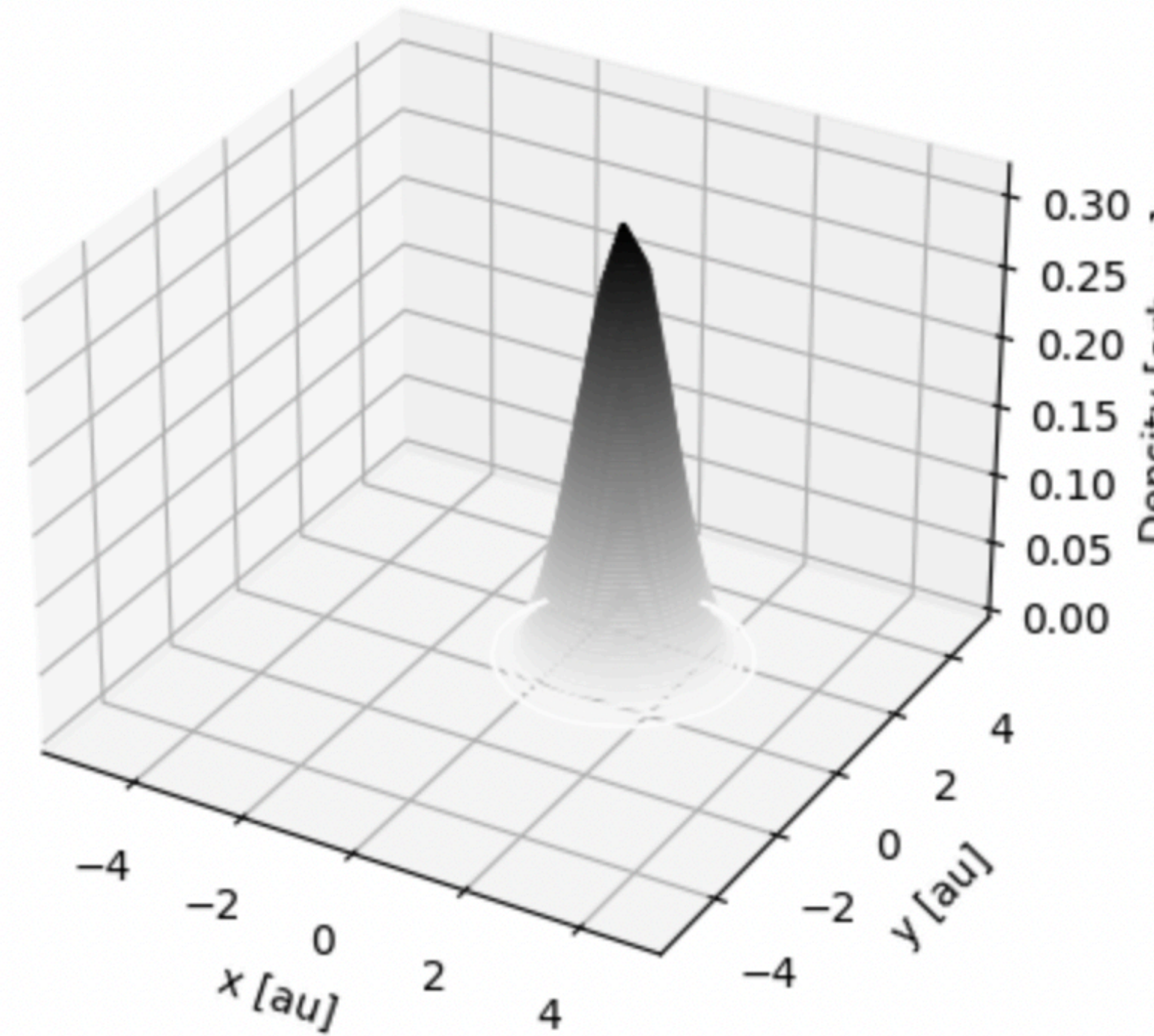
Clencheb

+ Section

+ Code + Text

Connect ^

```
ax = plt.axes(projection='3d')  
[ ] ax.contour3D(xmesh, ymesh, v_ft_psim, 100, cmap='binary')  
ax.set_xlabel('x [au]')  
ax.set_ylabel('y [au]')  
ax.set_zlabel('Density [arb. u.]');
```

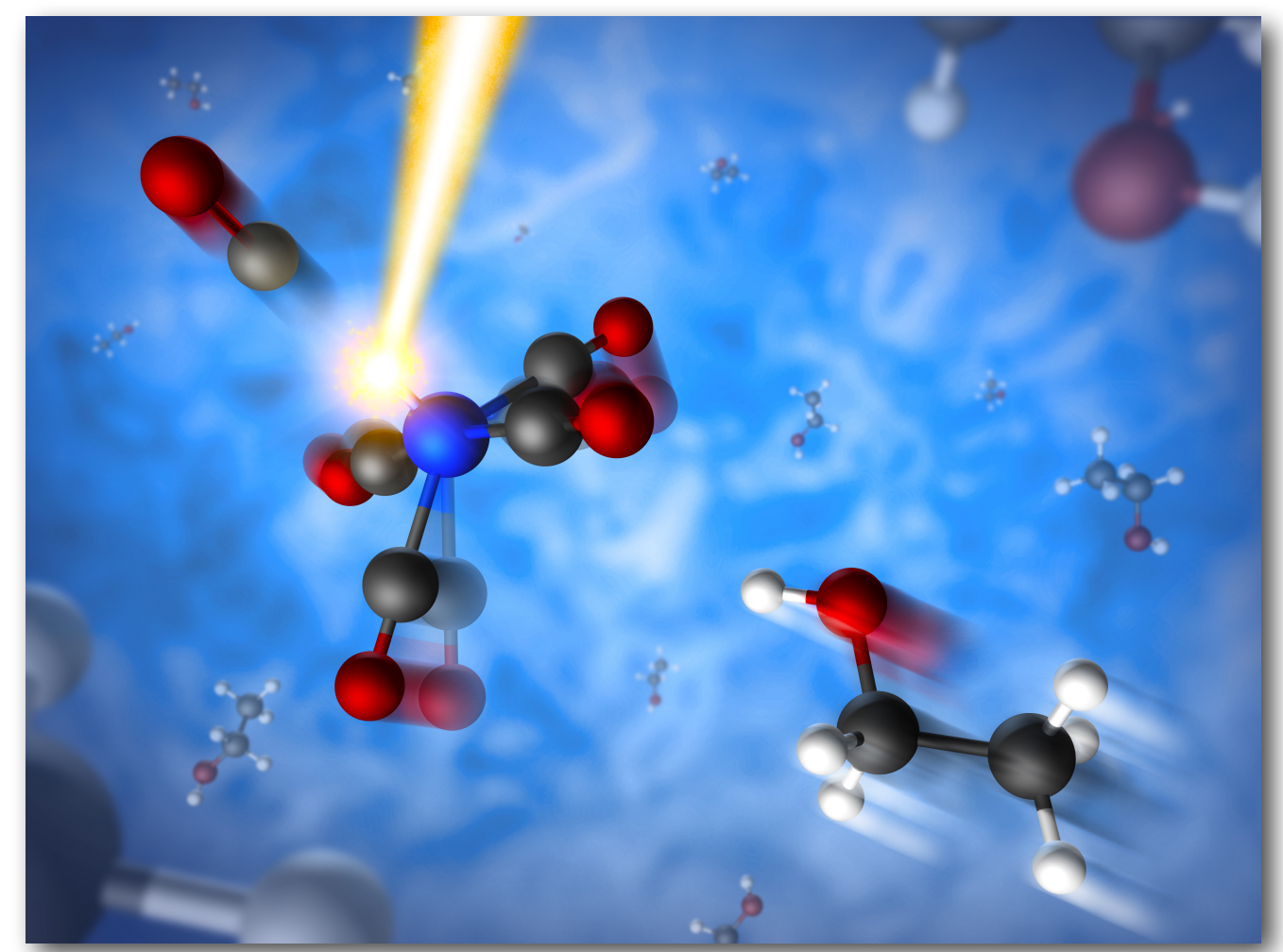
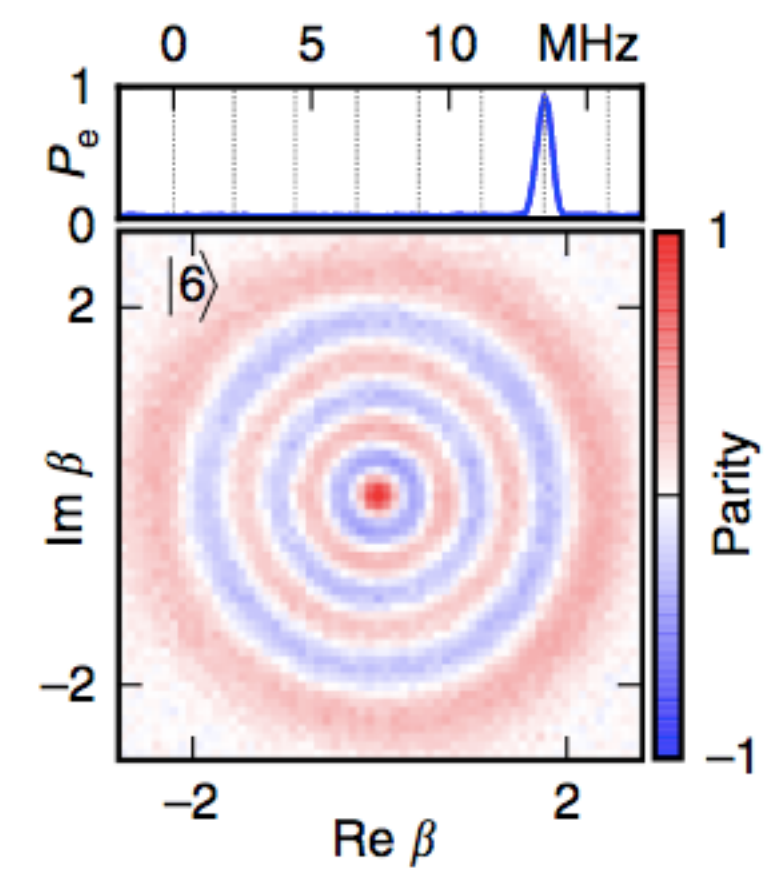
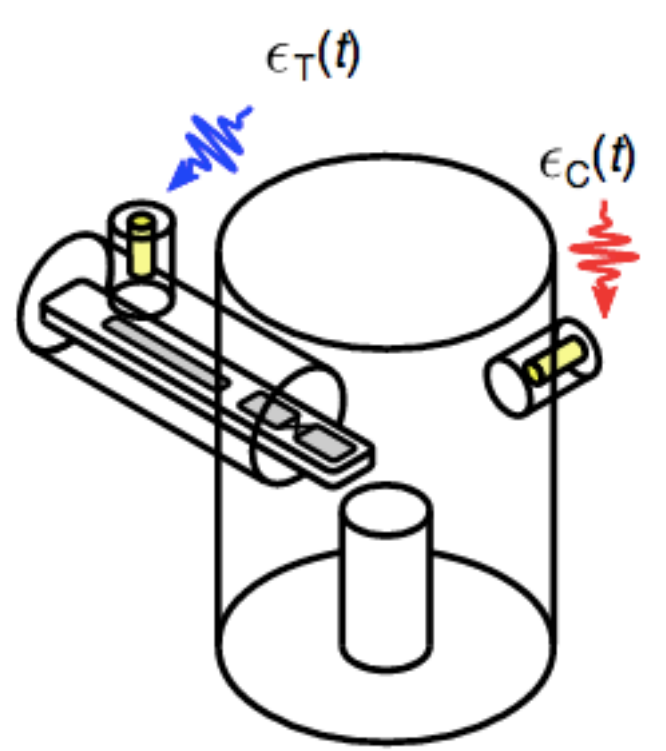
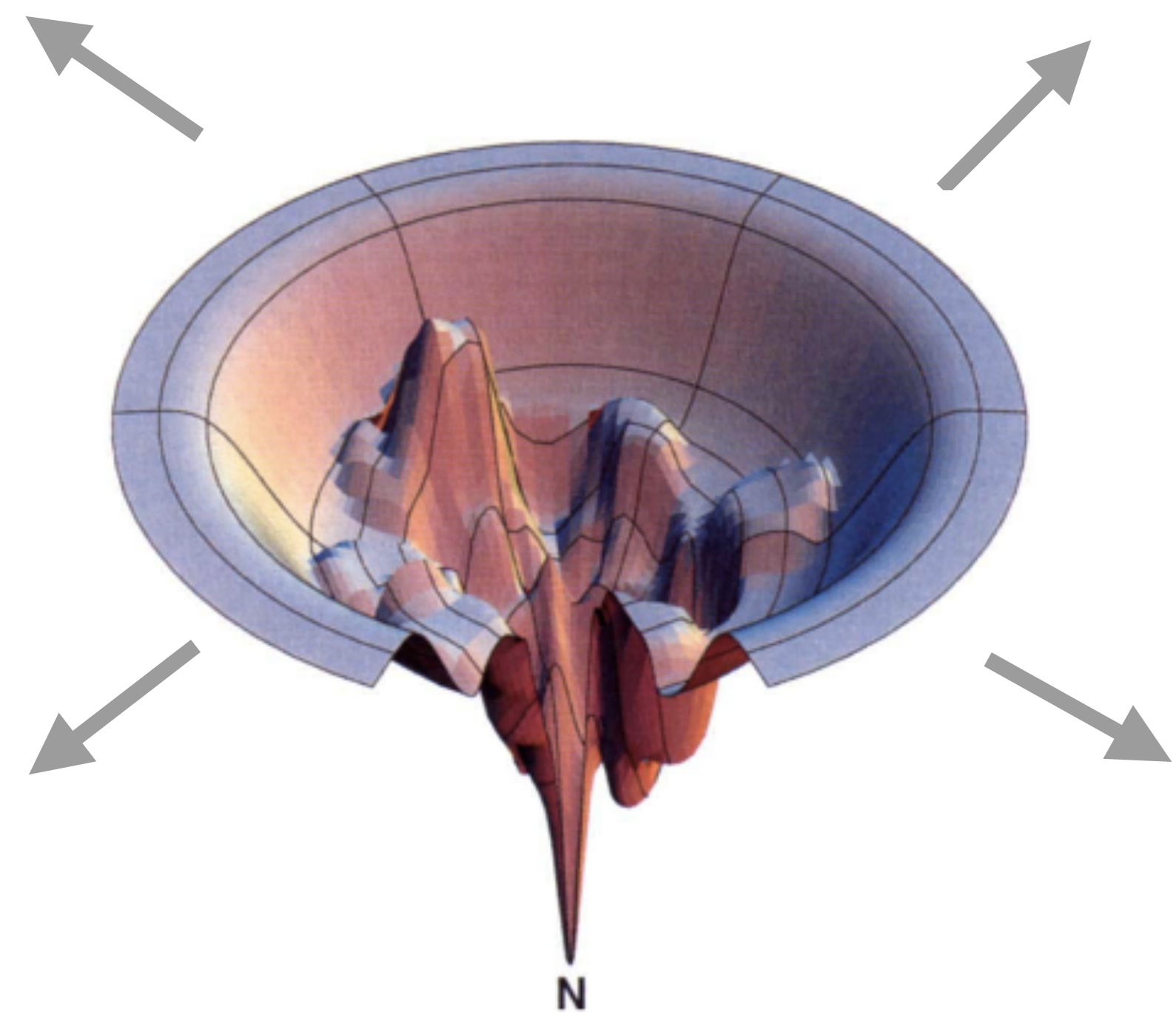
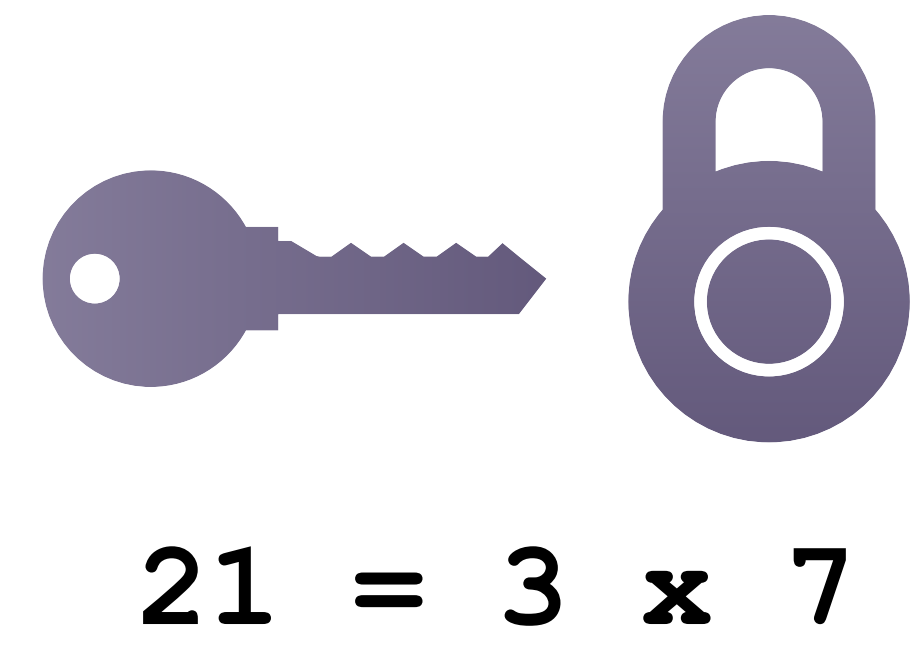
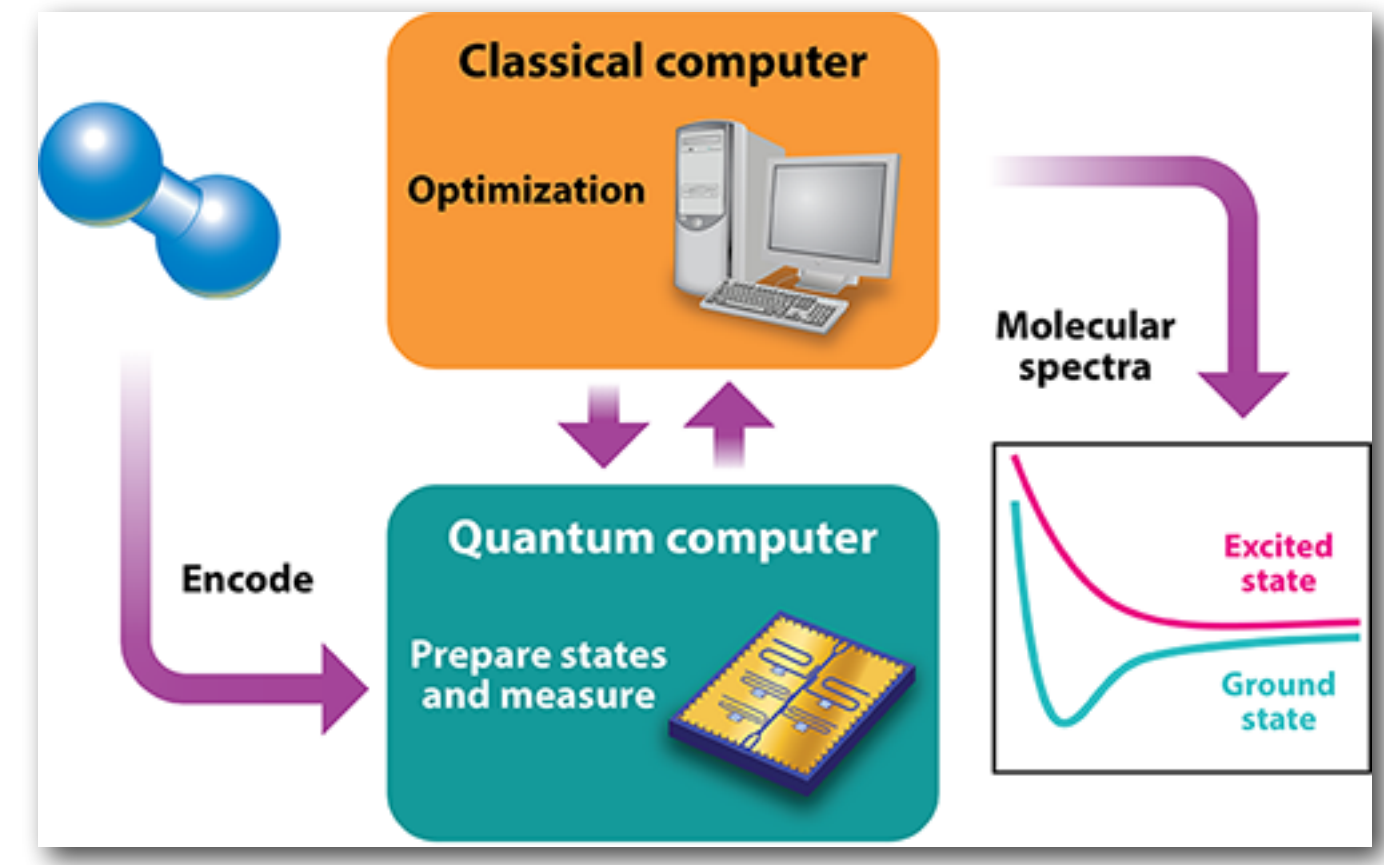


GLOBAL OPTIMIZATION WITH THE ITERATIVE POWER ALGORITHM

GLOBAL OPTIMIZATION

$$\arg \min_{x \in \mathbb{R}} V(x) = \{x^* \in \mathbb{R} \mid V(x) \geq V(x^*) \text{ for all } x \in \mathbb{R}\}$$

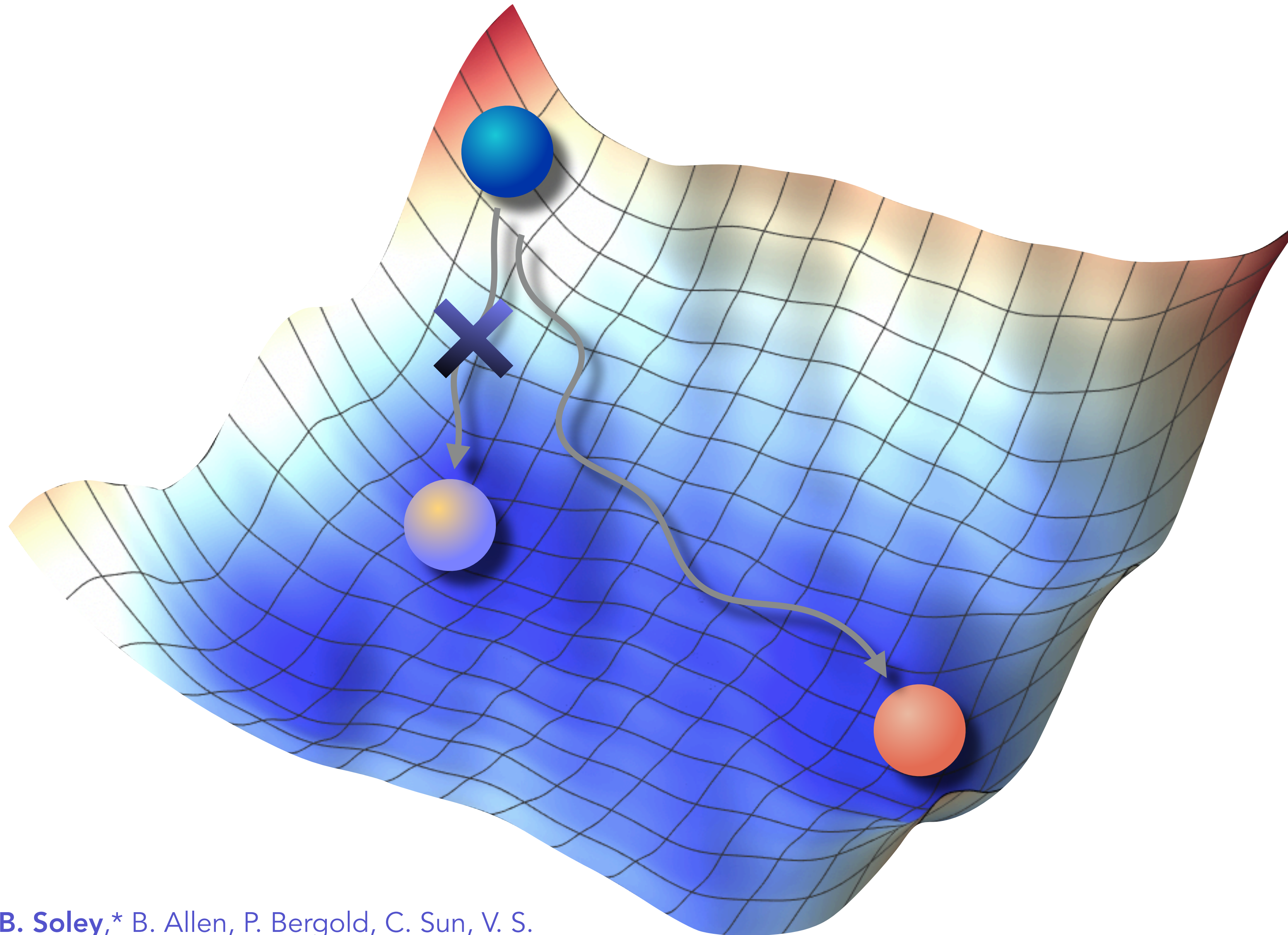
$V(\mathbf{x}) = \min!$ $V: \mathbb{R}^n \rightarrow \mathbb{R}$



S. Sim, J. Romero, P. D. Johnson, A. Aspuru-Guzik, *Physics* 11 (2018) 14.
 R. W. Heeres et al., *Nature Comm.* 8 (2017) 94.
 K. A. Dill, H. S. Chan, *Nat. Struct. Diol.* 4 (1997) 10.
 SLAC National Accelerator Laboratory.

M. B. Soley, P. Bergold, V. S. Batista, *JCTC* 17 (2021) 3280.
 T. H. Kyaw*, **Micheline B. Soley***, B. Allen, P. Bergold, C. Sun, V. S. Batista, A. Aspuru-Guzik, (2022) arXiv:2208.10470v1.
 M. B. Soley, A. Markmann, V. S. Batista, *JCTC*, 14 (2018) 3351.
 M. Soley, A. Markmann, V. S. Batista, *J. Phys. Chem. B*, 119 (2015) 715.
 K. A. Dill, J. L. MacCallum, *Science* 338 (2012) 1042.

LOCAL MINIMUM TRAPS

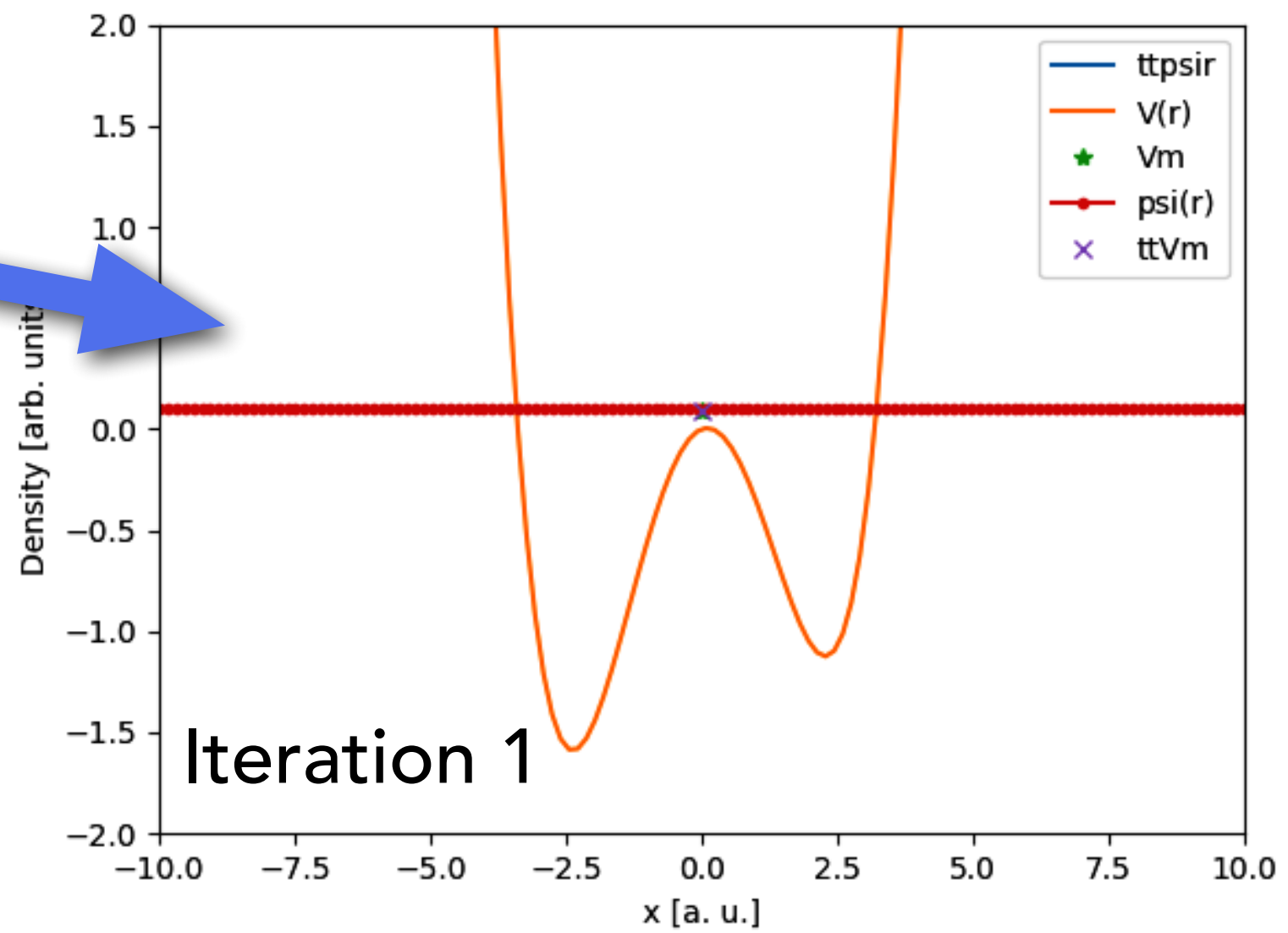


T. H. Kyaw*, **Micheline B. Soley***, B. Allen, P. Bergold, C. Sun, V. S. Batista, A. Aspuru-Guzik, (2022) arXiv:2208.10470v1.
M. B. Soley, P. Bergold, V. S. Batista, JCTC 17 (2021) 3280.

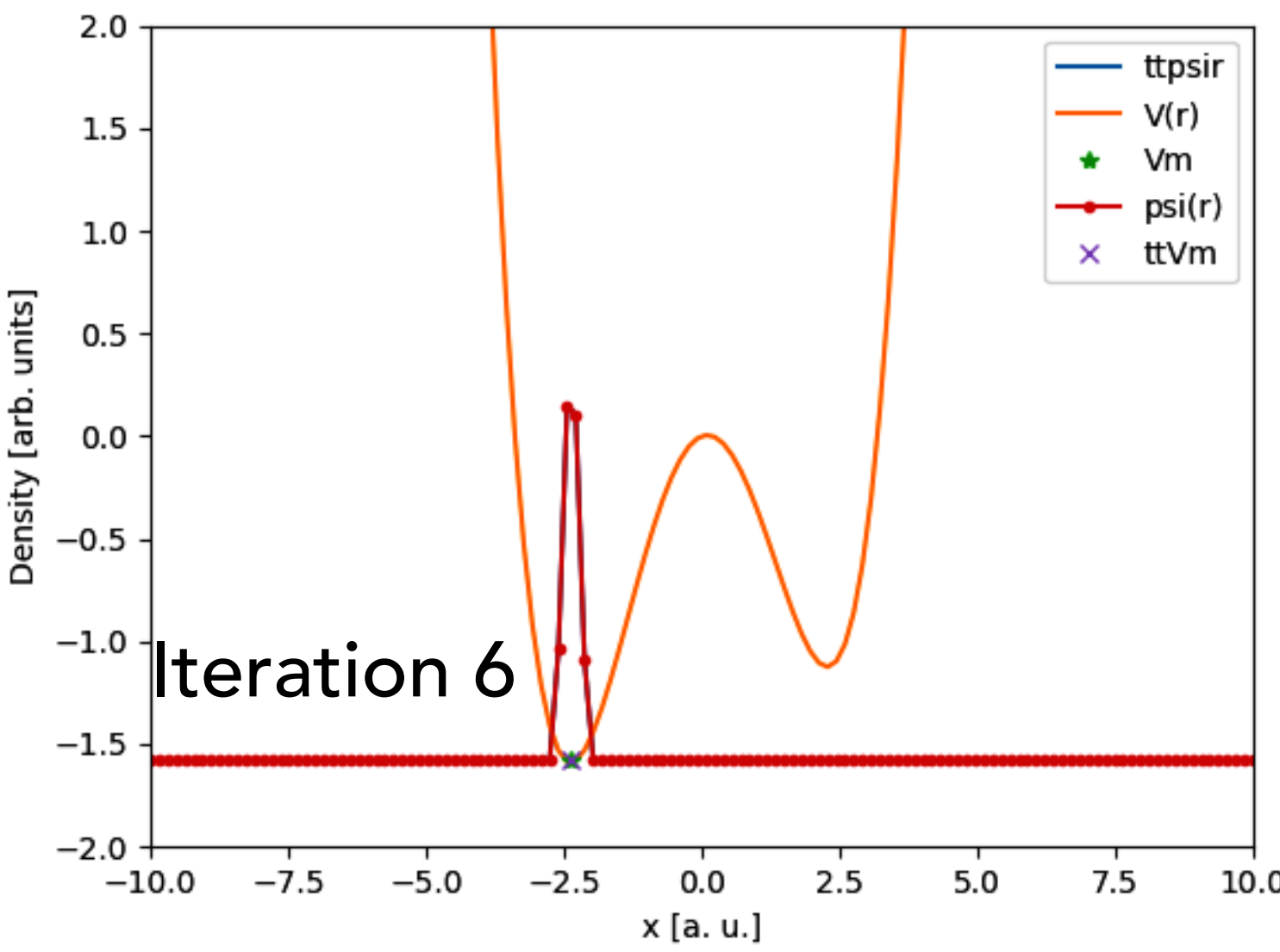
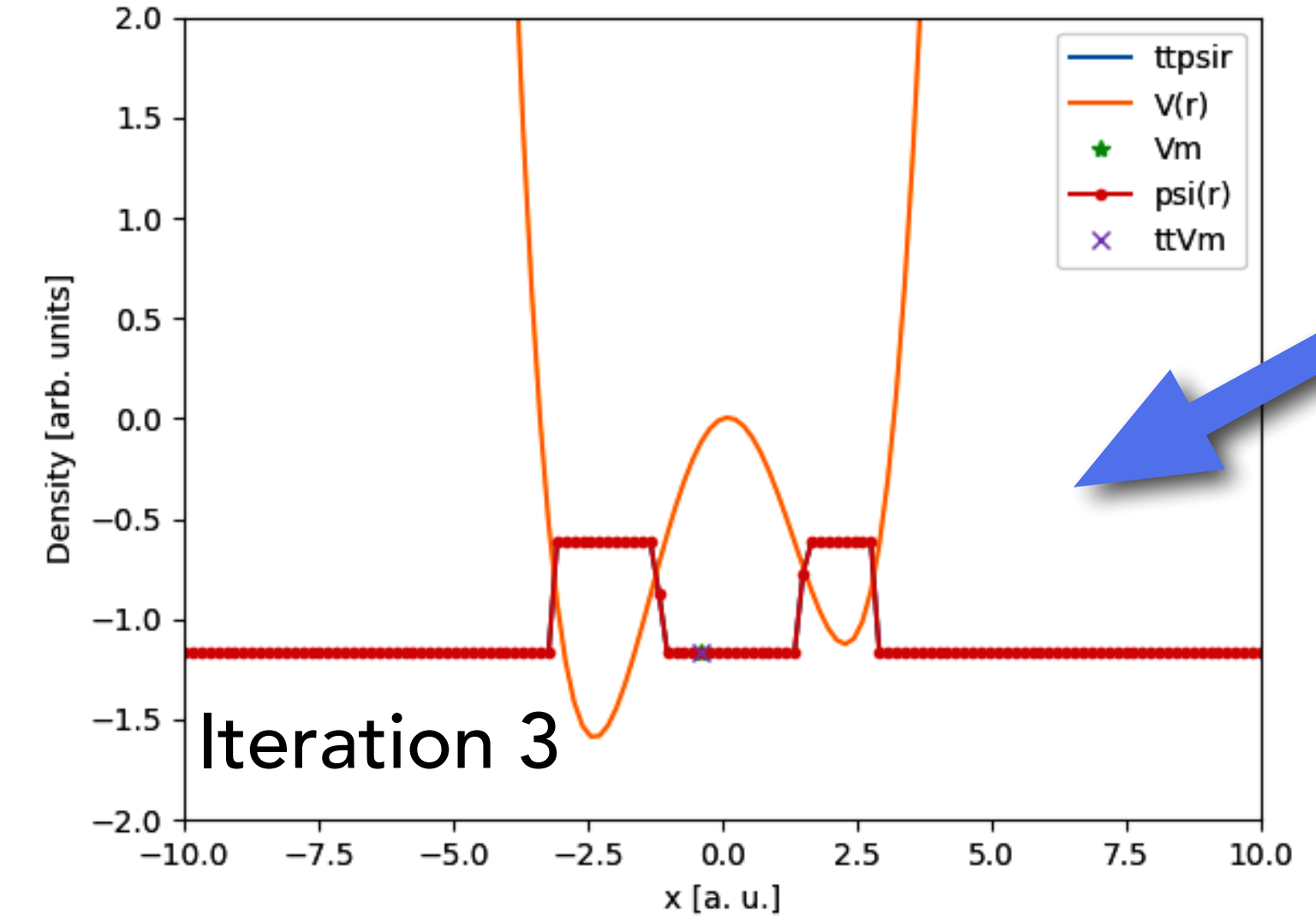
M. B. Soley, A. Markmann, V. S. Batista, JCTC, 14 (2018) 3351.
M. Soley, A. Markmann, V. S. Batista, J. Phys. Chem. B, 119 (2015) 715.

We introduce the **iterative power algorithm (IPA)** that bypasses local minimum traps to converge to the global minima.

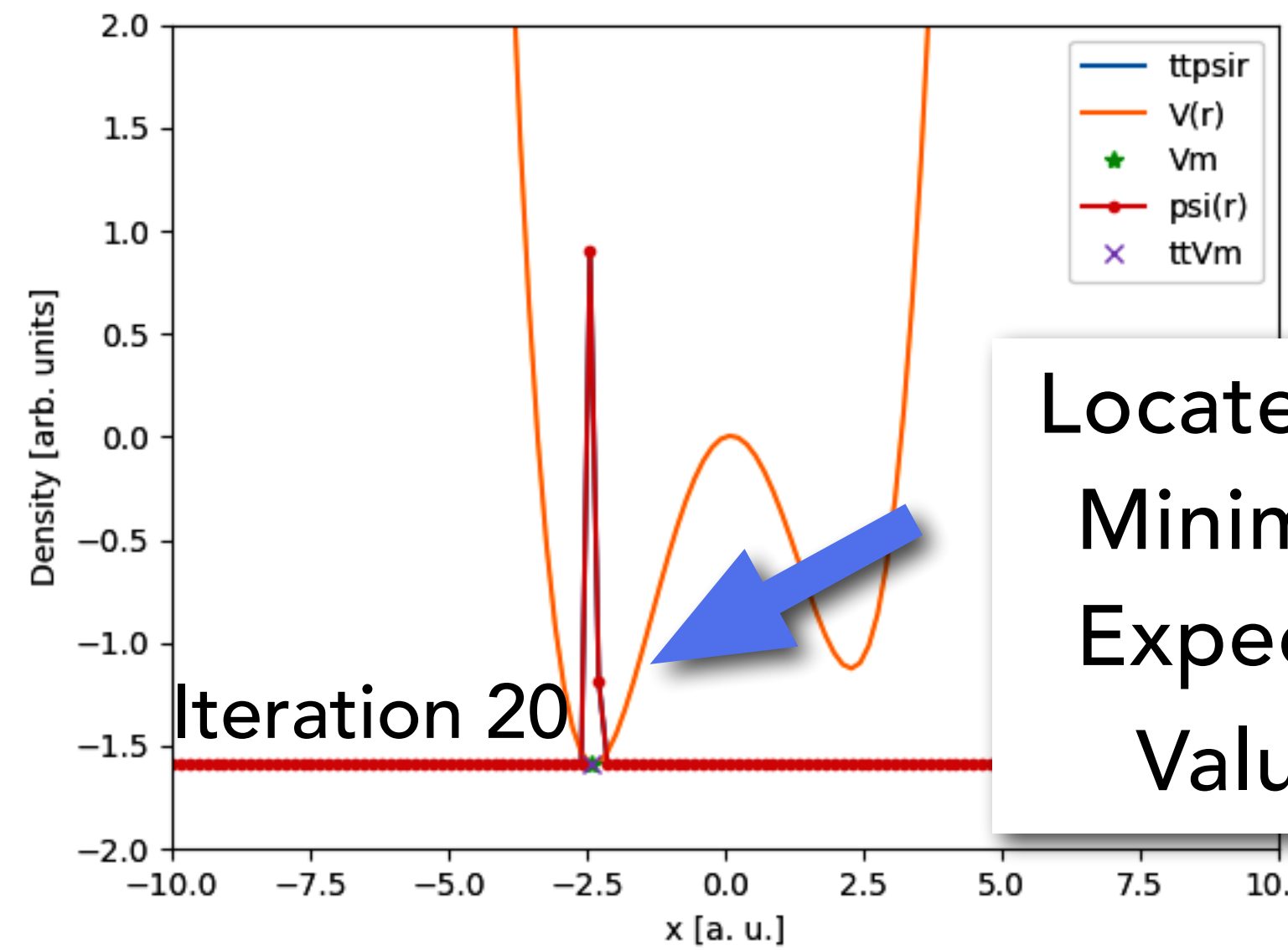
Initialize ρ and V as Tensor Trains



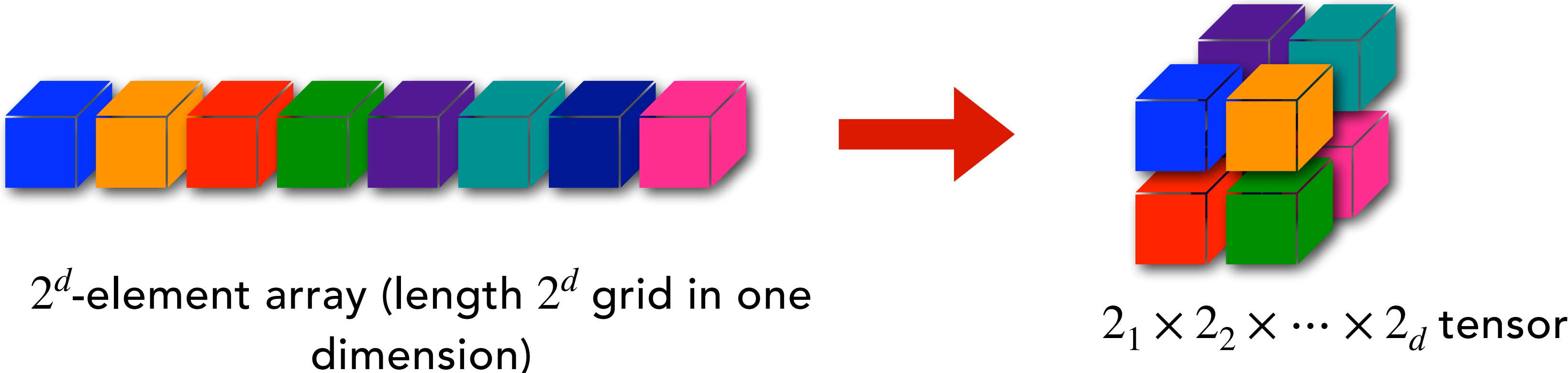
Iteratively Apply Oracle
 $\rho(x) \rightarrow \mathcal{N}U(x)\rho(x)$
 $U(x) = e^{-cV(x)}$



Locate Global Minimum as Expectation Value $\langle x \rangle$



IPA can efficiently find the global minimum of low-rank high-dimensional potential energy surfaces by approximating $\rho(\mathbf{x})$ and $V(\mathbf{x})$ in D physical dimensions in the form of **quantics tensor trains (QTTs)** in n reshaped dimensions.



QTT

$$\approx \sum_{\alpha_1=1}^{r_1} \sum_{\alpha_2=1}^{r_2} \dots \sum_{\alpha_{d-1}=1}^{r_{d-1}} A_1(1, i_1, \alpha_1) A_2(\alpha_1, i_2, \alpha_2) \dots A_d(\alpha_{d-1}, i_d, 1)$$

THE IPA METHOD

INITIALIZATION: UNIFORM SUPERPOSITION

- Consider the cost function as a potential surface $V(\mathbf{x})$ with global minimum at $\mathbf{x} = \mathbf{x}^\star$
- Initialize a probability distribution $\rho_0(\mathbf{x})$ in the potential as a quantics tensor train

$$\rho_0 : \mathbb{R} \rightarrow [0, \infty)$$

$$\|\rho_0\|_{L^1} = \int_{\mathbb{R}} dx \rho_0(x) = 1$$

$$\int_{x^\star - r}^{x^\star + r} dx \rho_0(x) > 0 \quad \text{where } r > 0$$

EVOLUTION: AMPLITUDE AMPLIFICATION

DEFINE $U(x)$

(i) a continuous and strictly positive function maximized at the global minima of $V(x)$

$$\arg \max_{x \in \mathbb{R}} U(x) = \arg \min_{x \in \mathbb{R}} V(x)$$

(ii) an integrable function ($U(x) \in L^1(\mathbb{R})$)

EXAMPLE

$$U(x) = e^{-\beta V(x)} \text{ for fixed scaling parameter } \beta > 0$$

ITERATIVELY APPLY RECURRENCE RELATION

for $k = 1, 2, \dots$

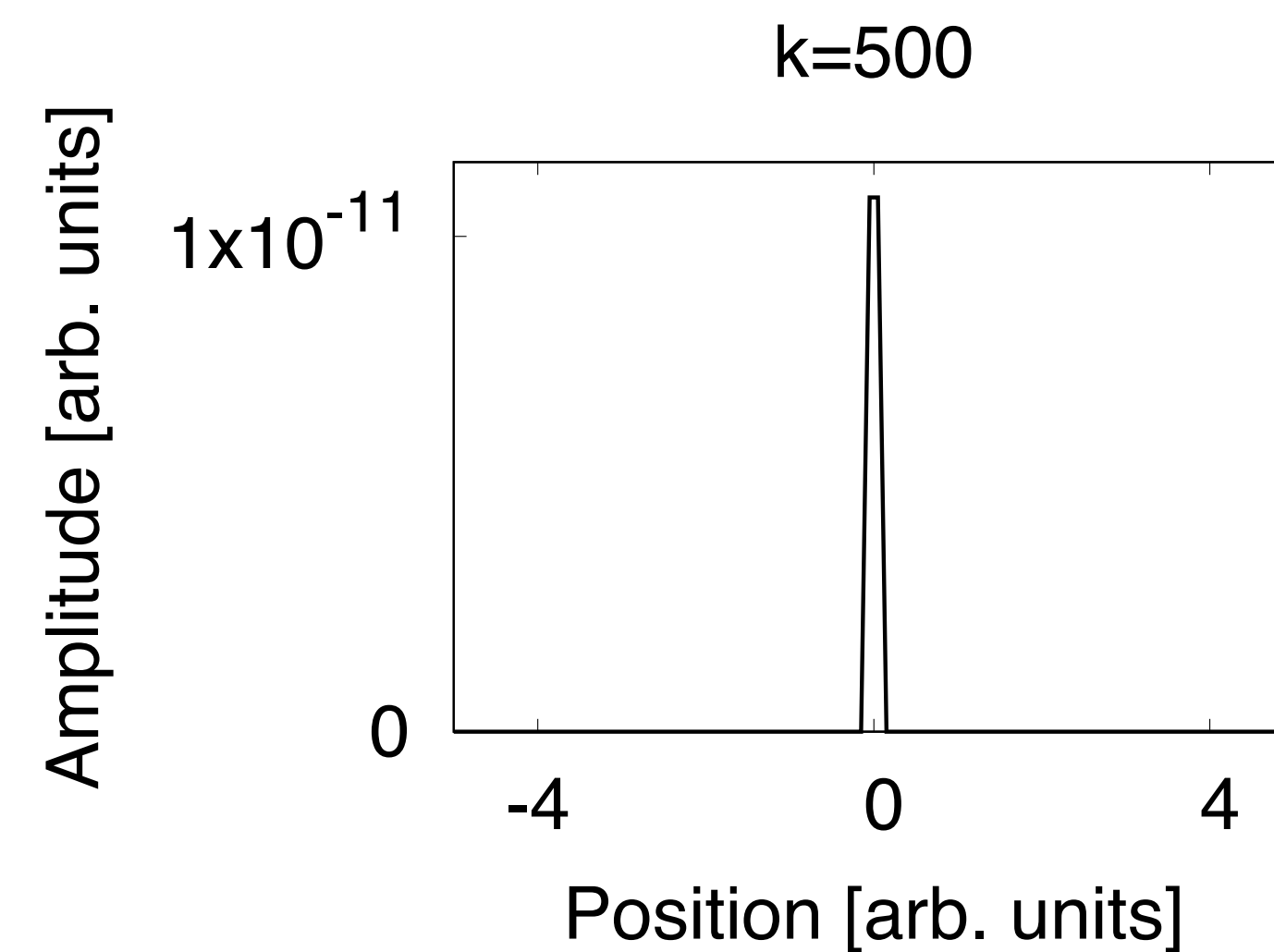
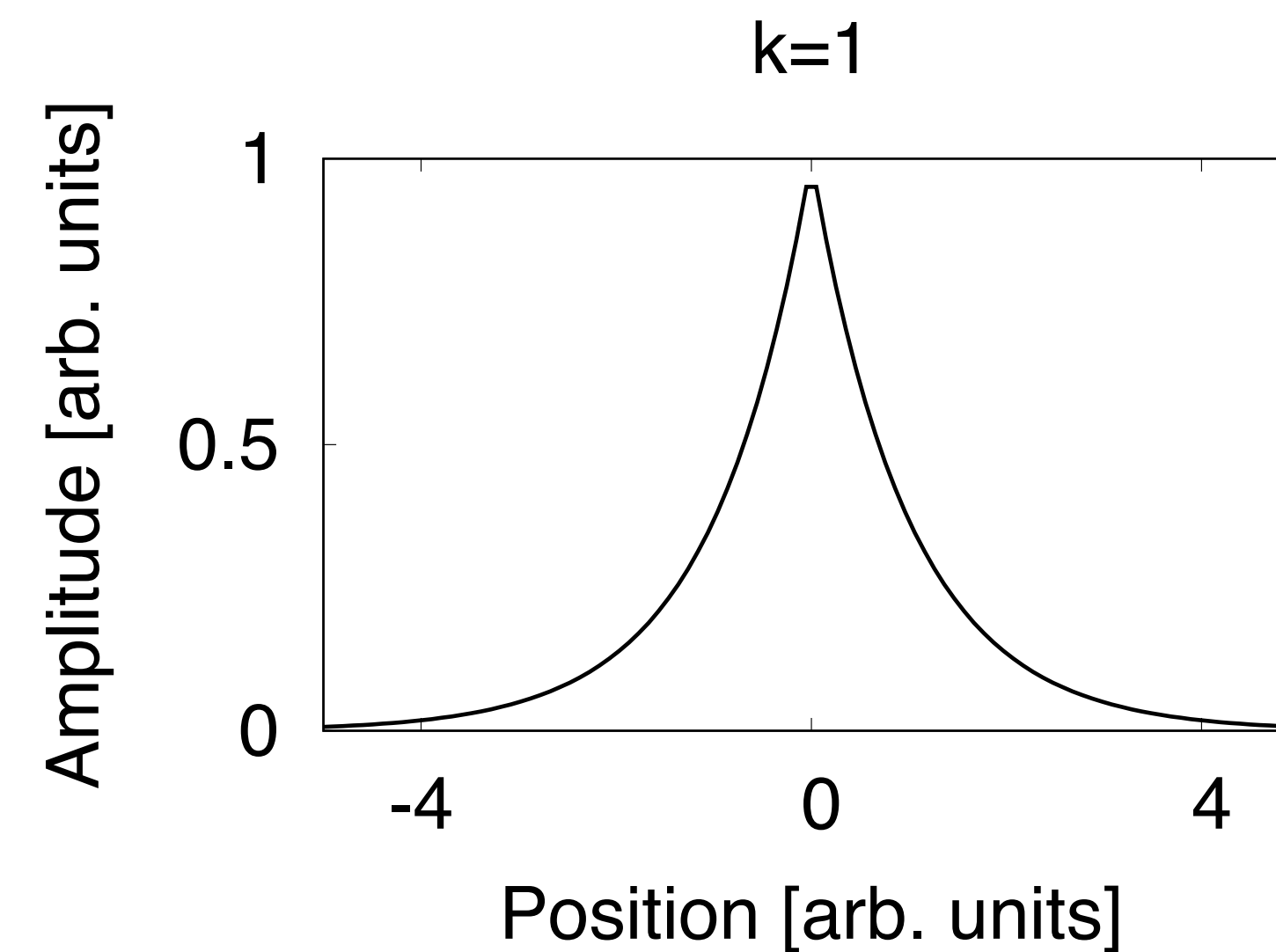
$$\eta_k = \|U\rho_{k-1}\|_{L^1} = \int_{\mathbb{R}} dx U(x)\rho_{k-1}(x);$$

$$\rho_k(x) = \frac{U(x)\rho_{k-1}(x)}{\eta_k} = \frac{(U(x))^k \rho_0(x)}{\|U^k \rho_0\|_{L^1}};$$

end

$$\rho_{\text{final}}(x) = \lim_{k \rightarrow \infty} \rho_k(x) = \sum_{j=1}^s \delta(x - x_j^{\star})$$

U^k for $U = e^{-V(x)}$ with $V(x) = |x|$



RESOLUTION OF GLOBAL MINIMA: MEASUREMENT

OBTAIN THE GLOBAL MINIMUM POSITION

(i) Single global minimum

$$x^{\star} = \langle x \rangle_{\rho_{\text{final}}} = \int_{\mathbb{R}} dx x \rho_{\text{final}}(x)$$

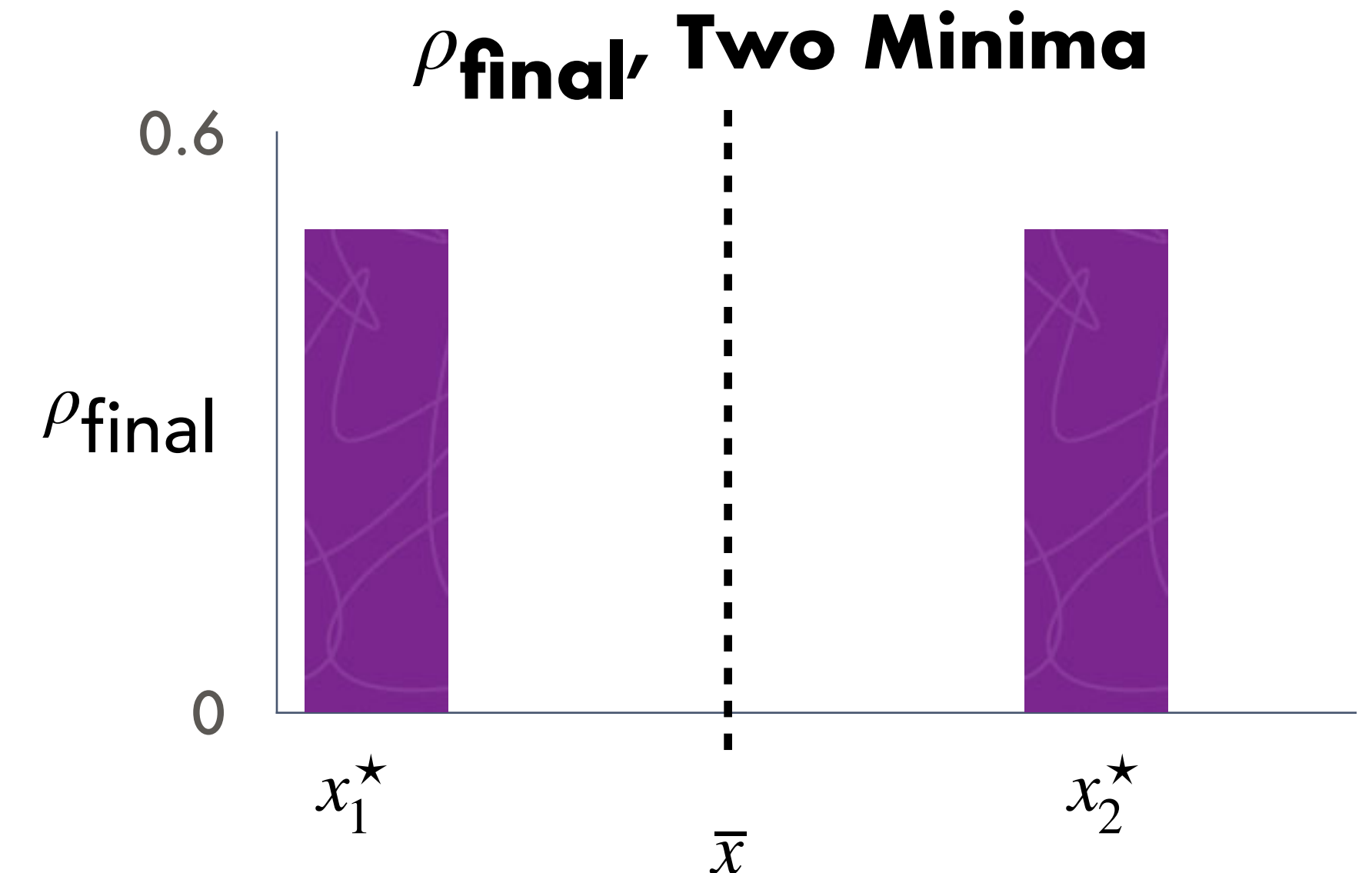
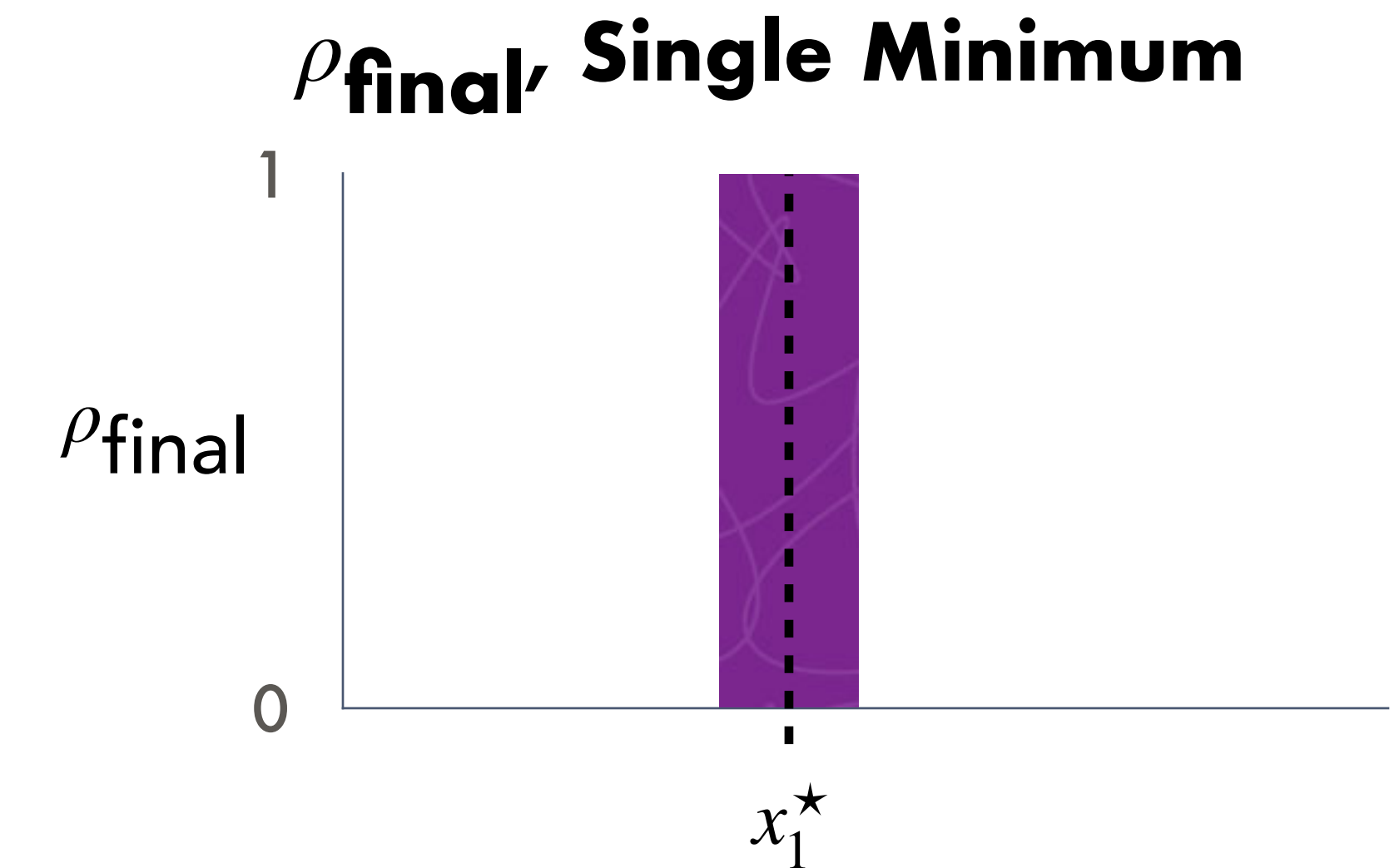
(ii) Two degenerate global minima

$$\bar{x} = \langle x \rangle_{\rho_{\text{final}}}$$

$$\Theta(x - \bar{x}) = \begin{cases} 0, & \text{if } x \leq \bar{x} \\ 1, & \text{if } x > \bar{x} \end{cases}$$

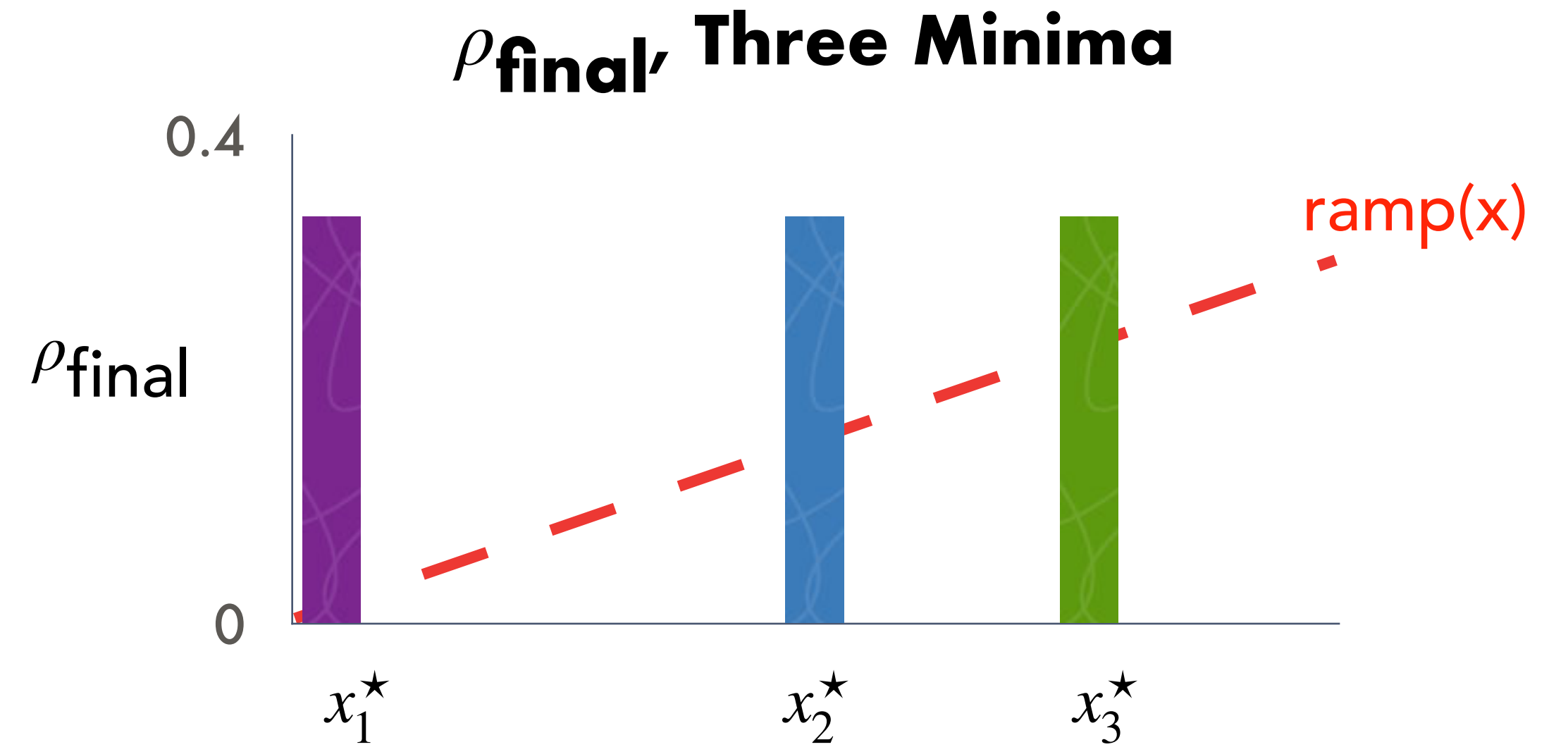
$$x_1^{\star} = \langle x \rangle_{\rho_{\text{final}}}(x) \Theta(x - \bar{x})$$

$$x_2^{\star} = \langle x \rangle_{\rho_{\text{final}}}(x) (1 - \Theta(x - \bar{x}))$$

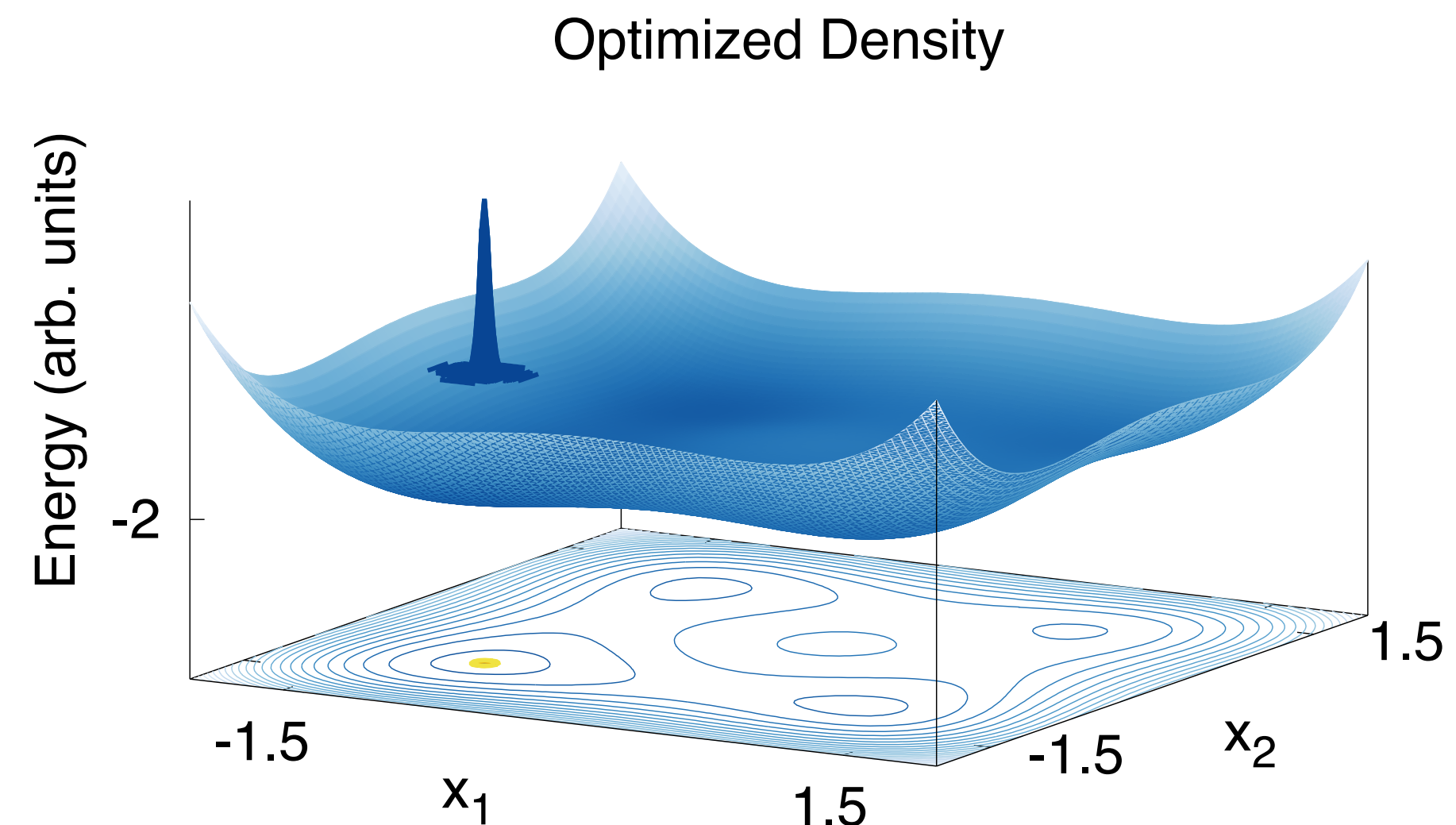
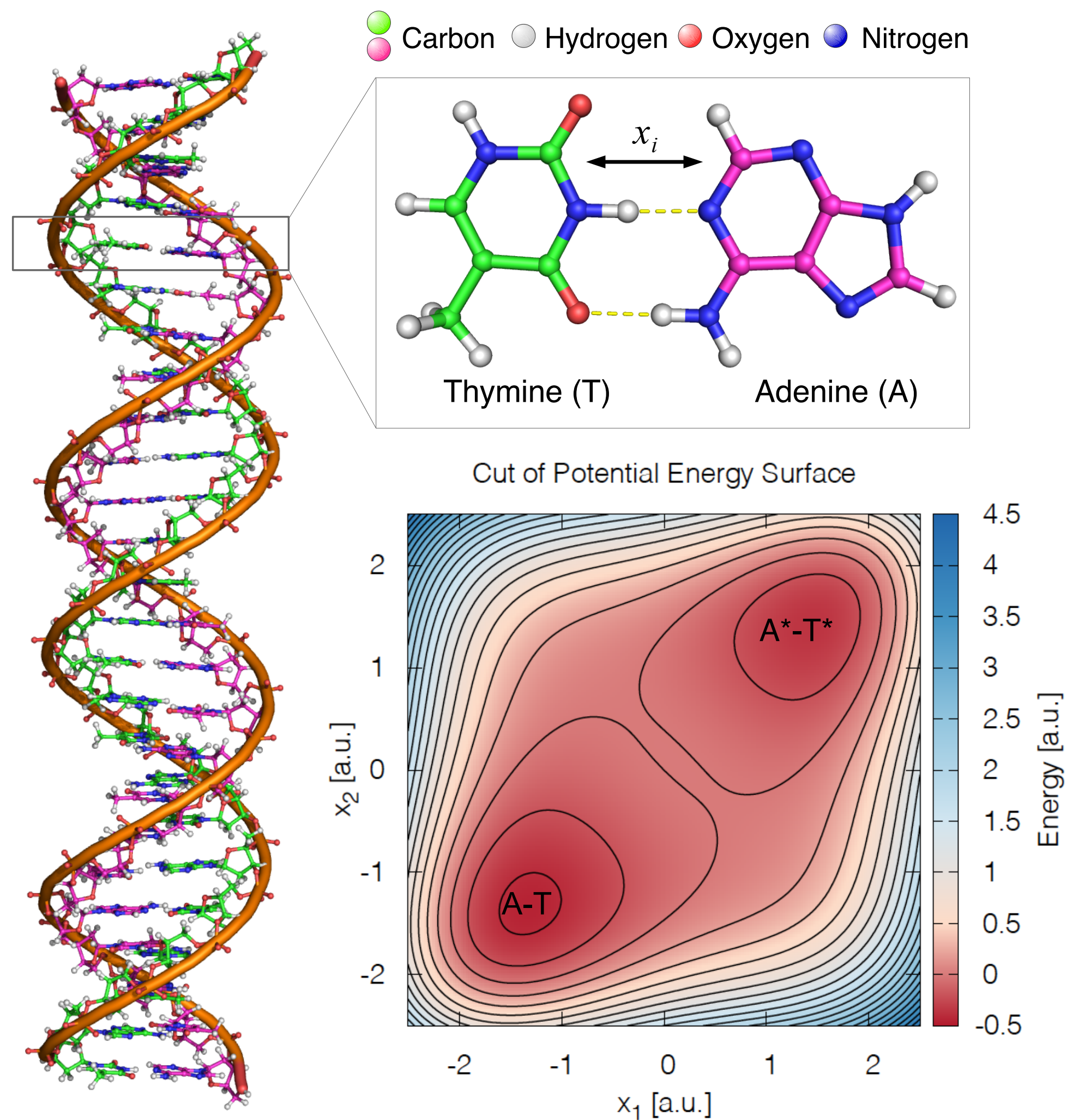


(iii) Unknown number of global minima

- Obtain ρ_{final} using IPA
- Reinitialize $\rho_0 \propto \rho_{\text{final}}$
- Isolate first component x_1^* of Dirac comb with a second use of IPA using a "ramp potential" (e.g. $\text{ramp}(x) = x$)
- Multiply ρ_{final} by the Heaviside function $\Theta(x - x_1^*)$
- Repeat process to identify the remaining components x_i^* until all global minima are resolved



IPA FOR HYDROGEN BOND CONFIGURATIONS



IPA successfully identifies global optimization of functions with up to 2^{50} local minima, beyond the capabilities of a straightforward enumeration approach.

IPA CONVERGENCE

Number of iterations required to reach optimal result with 50% certainty (à la Grover's algorithm):

$$U = \text{diag}(\lambda_2, \dots, \lambda_2, \lambda_1, \lambda_2, \dots, \lambda_2) \in \mathbb{R}^{n \times n}, \quad 0 < \lambda_2 < \lambda_1$$

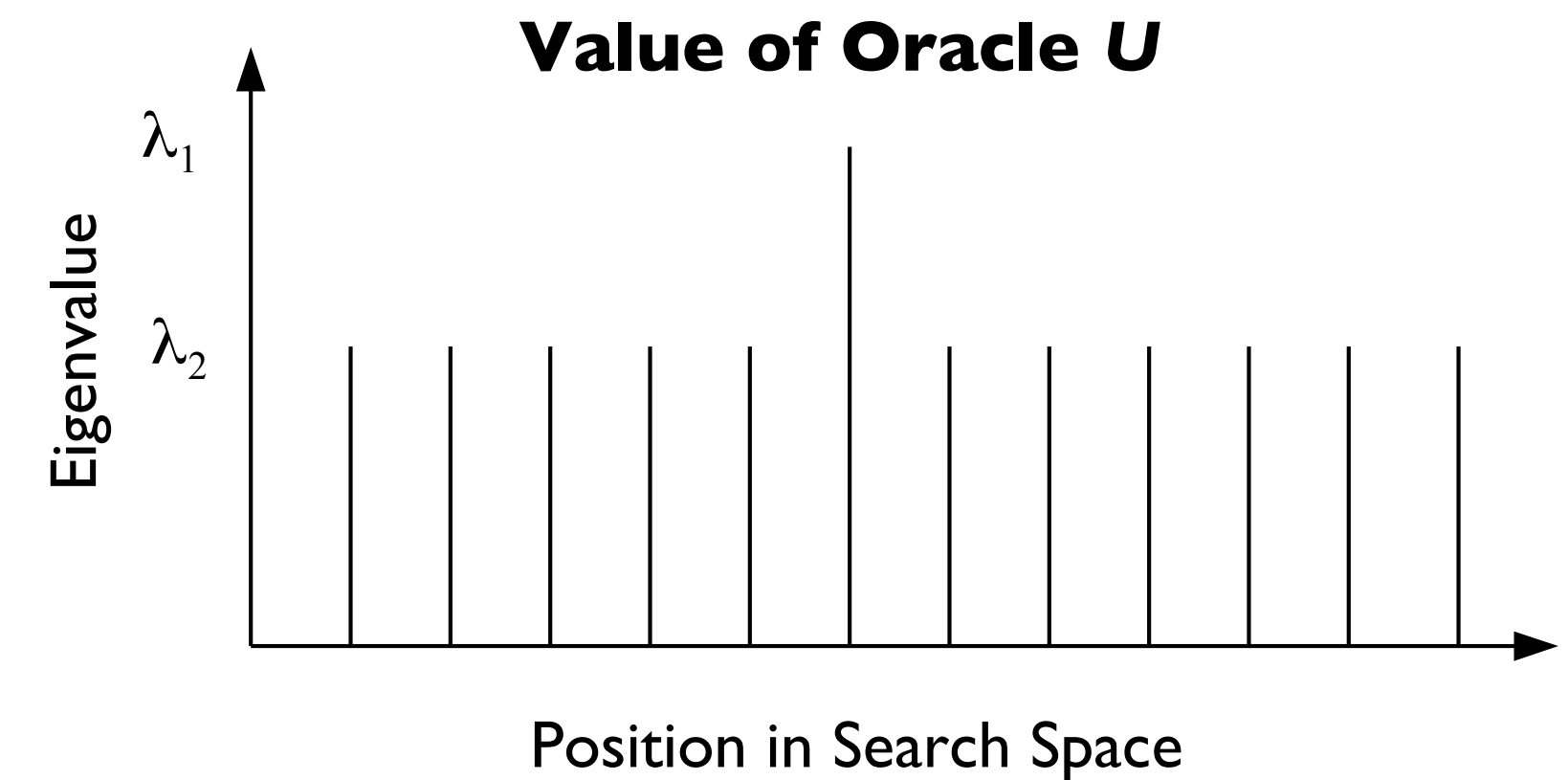
$$\rho_0 = \frac{1}{n}(1, \dots, 1) \in \mathbb{R}^n$$

$$\rho_k = \frac{U^k \rho_0}{\|U^k \rho_0\|_1} \quad \frac{\rho_{k, \min}}{\rho_{k, \max}} = \left(\frac{\lambda_2}{\lambda_1} \right)^k$$

$$1 = \|\rho_k\| = \rho_{k, \max} + (n - 1)\rho_{k, \min}$$

$$\rho_{k, \max} = \frac{1}{1 + (n - 1) \cdot (\lambda_2/\lambda_1)^k}$$

$$\frac{1}{2} \leq \frac{1}{1 + (n - 1) \cdot (\lambda_2/\lambda_1)^k}$$



$$k \geq \frac{\log(n - 1)}{\log(\lambda_1/\lambda_2)}$$

IPA requires fewer iterations than foremost quantum approach.

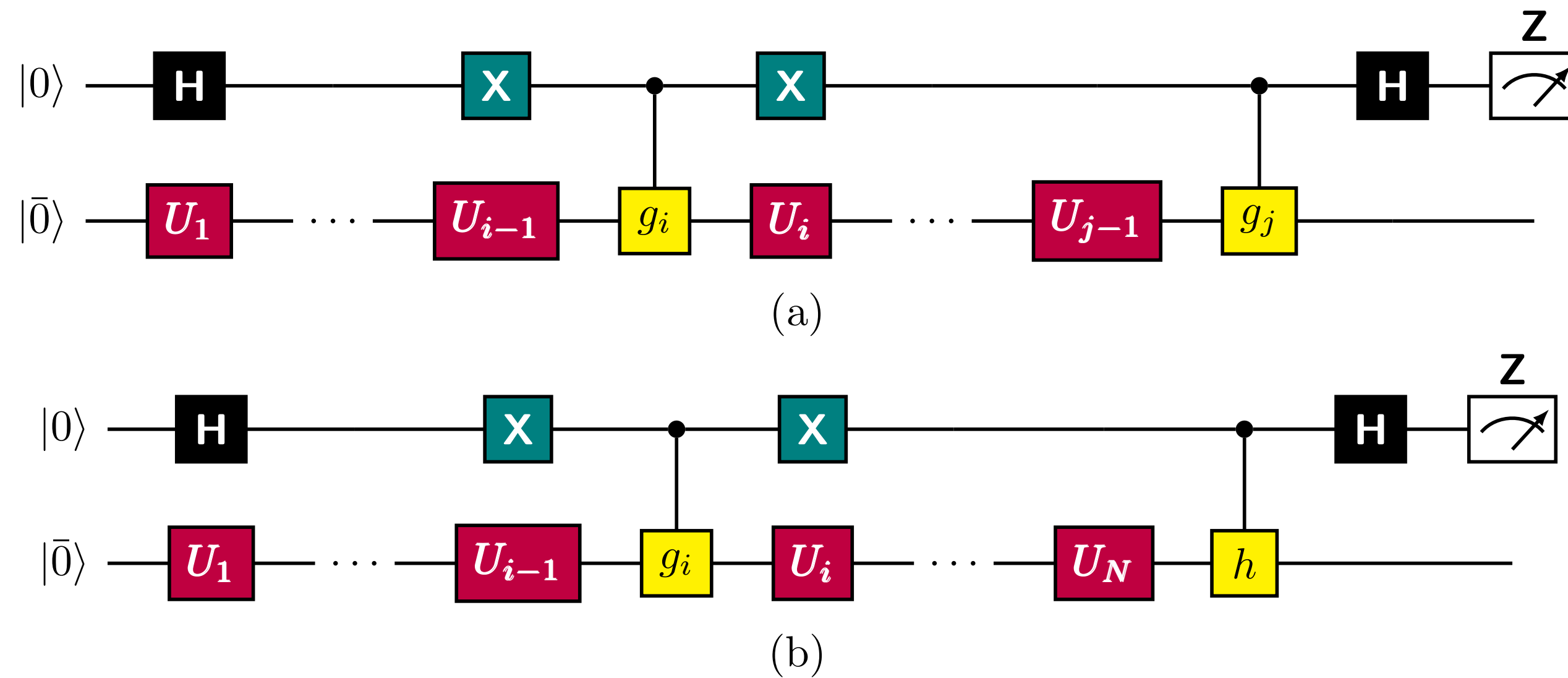
Micheline B. Soley, P. Bergold, V. S. Batista, JCTC 17 (2021) 3280.

L. K. Grover, Proceedings, 28th Annual ACM Symposium on the Theory of Computing, May 1996, 212.

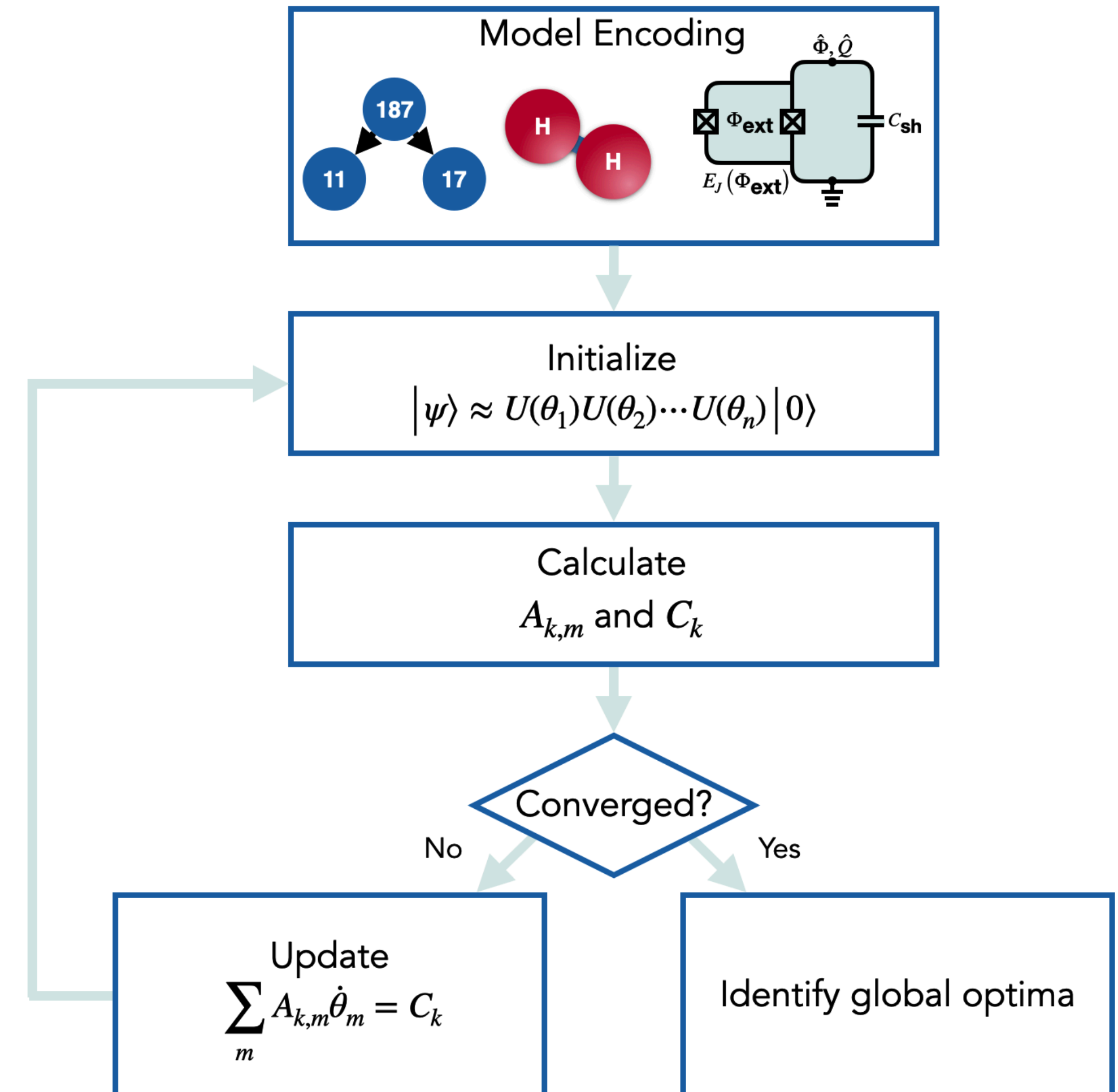
QUANTUM ITERATIVE POWER ALGORITHM (QIPA)

The McLachlan variational principle enables IPA to be also performed as a hybrid quantum-classical algorithm.

Quantum Circuits for $A_{k,m}$ and C_k

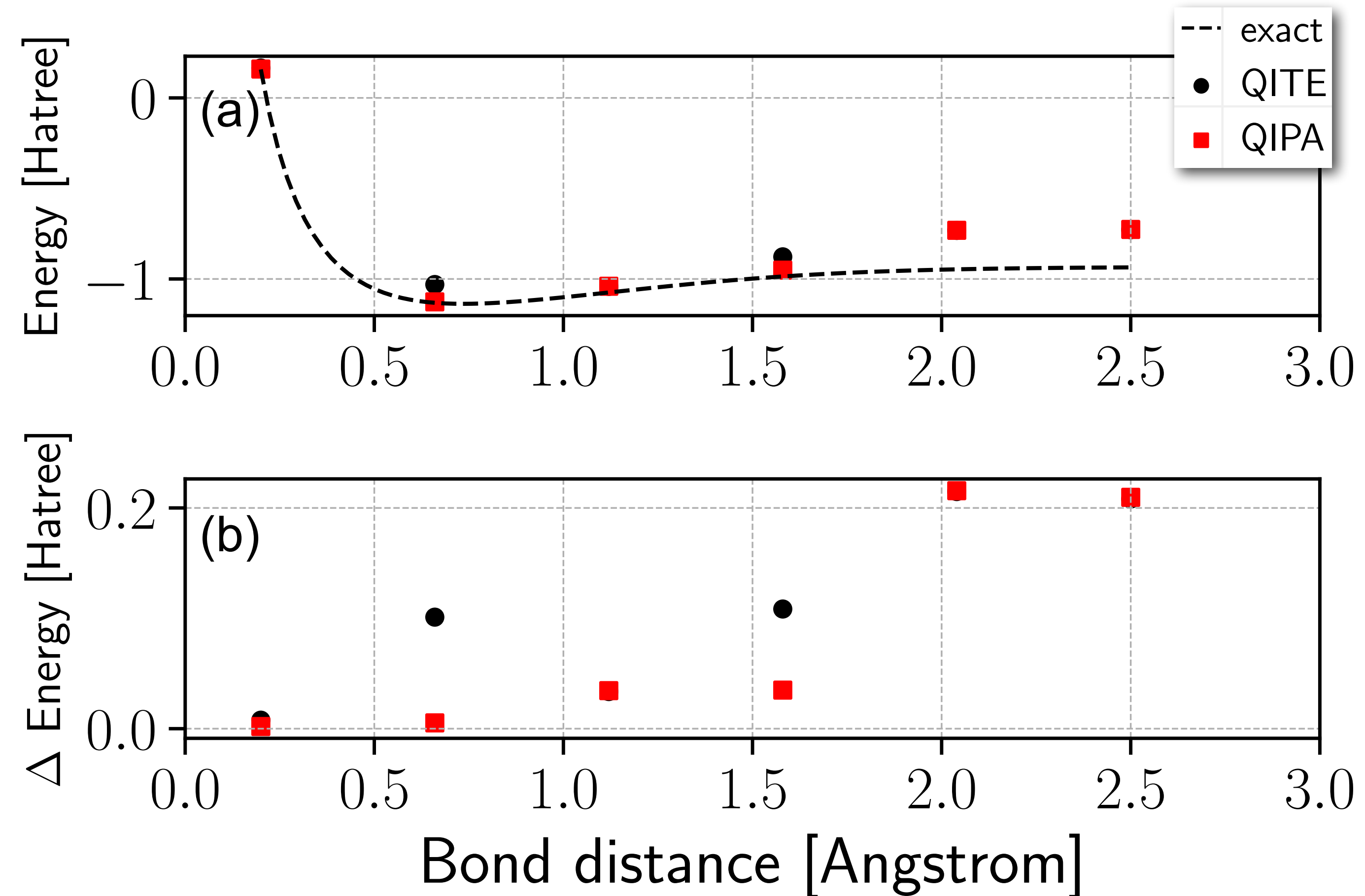
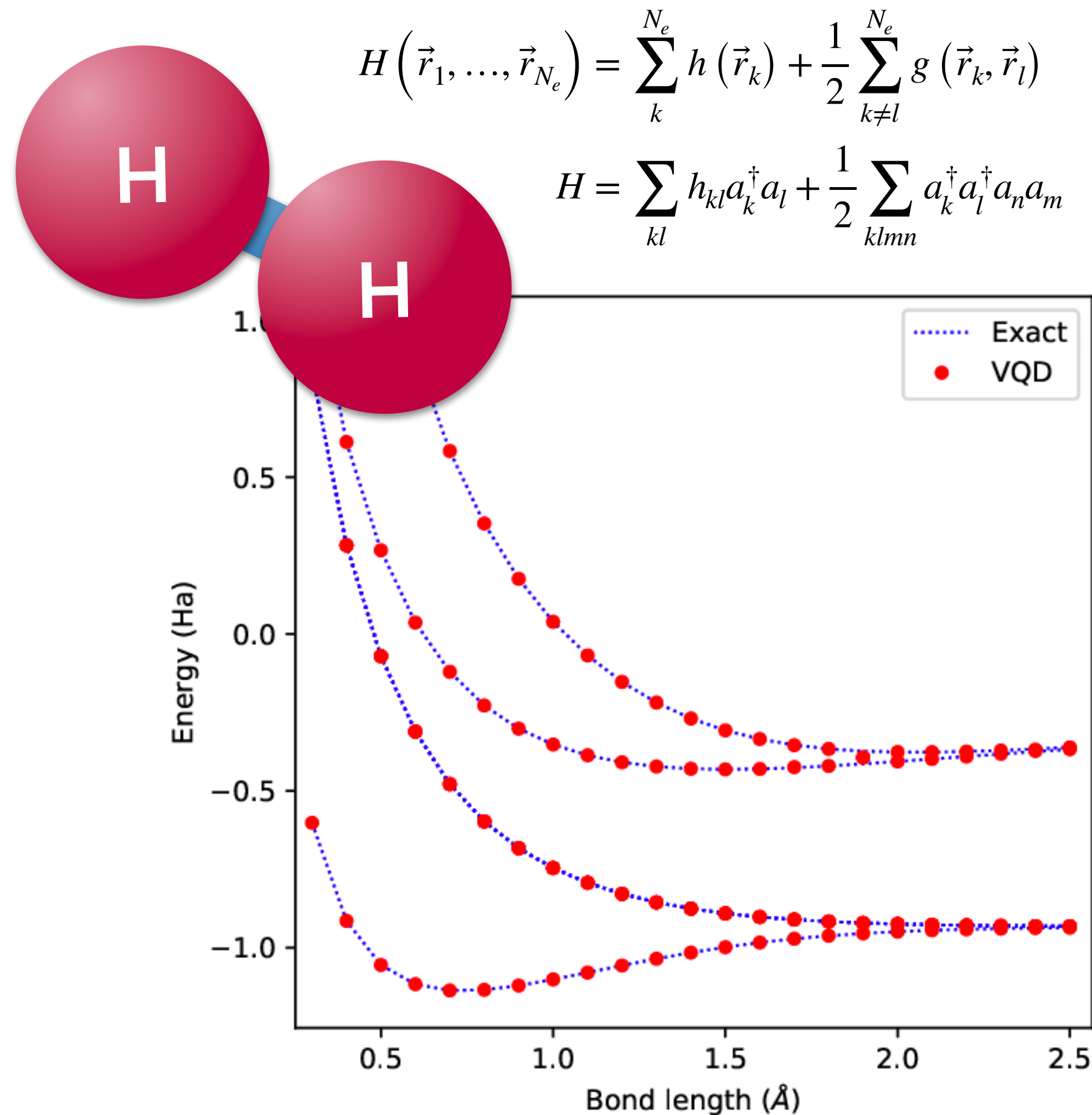


Scheme



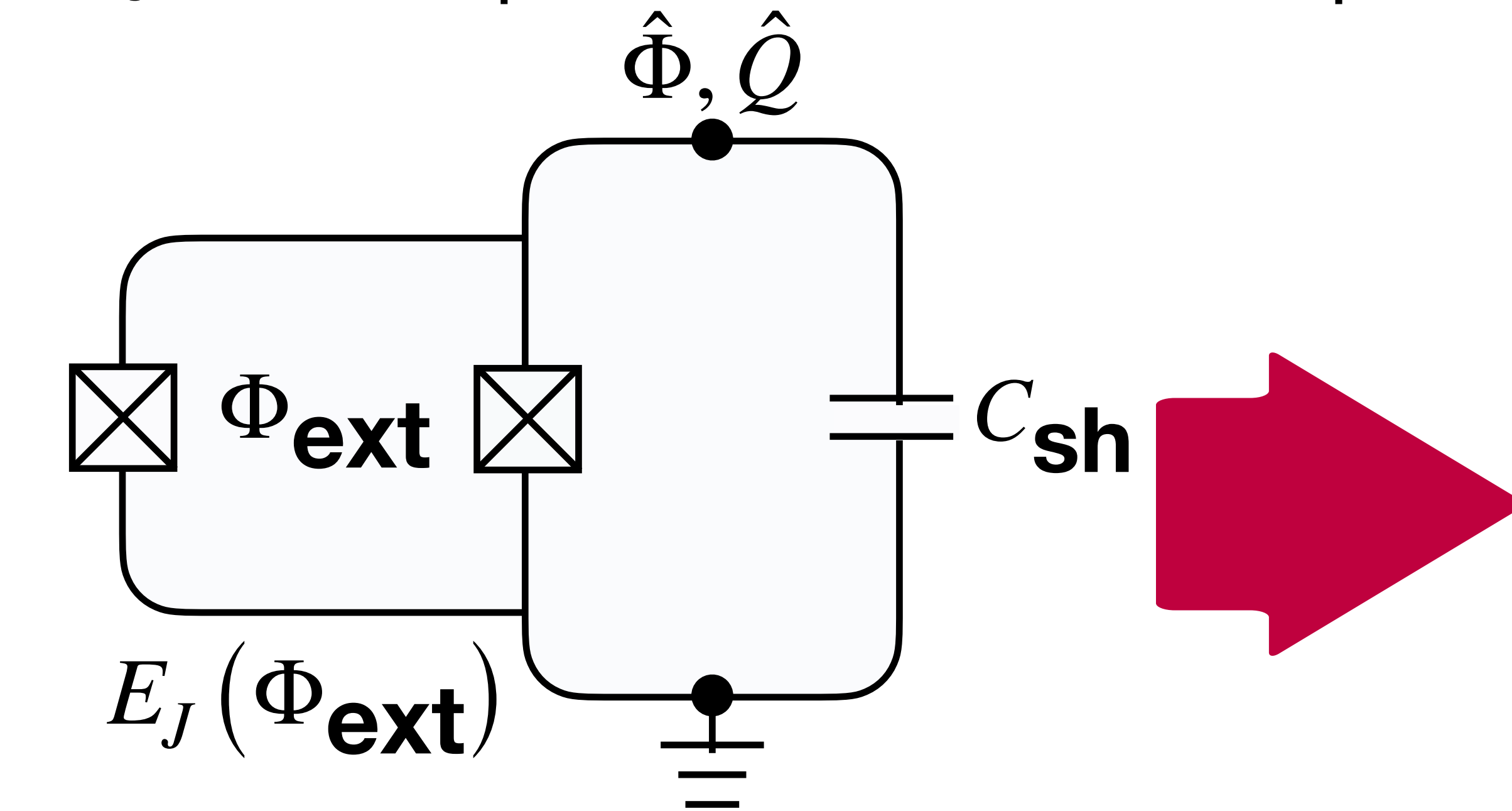
QIPA FOR ELECTRONIC STRUCTURE THEORY

QIPA **successfully** identifies the ground state energies of H₂ to high accuracy

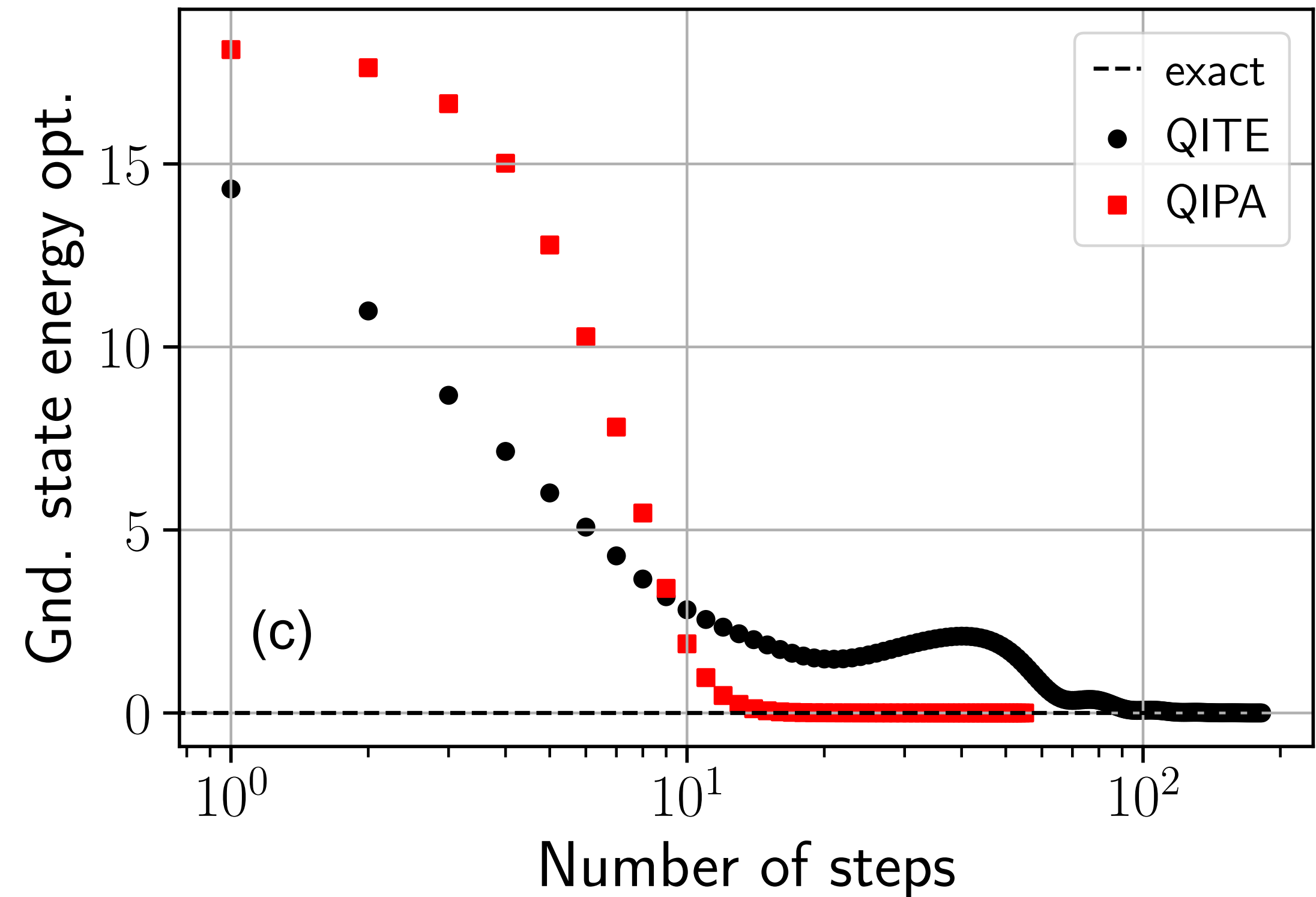


QIPA FOR QUANTUM COMPUTER DESIGN

First dynamical optimization of quantum processor design via quantum computing



$$\hat{H}_{M\text{-transmon}} = 2e^2 \sum_{i,j=1}^M (\mathbf{C}^{-1})_{ij} \hat{N}_i \hat{N}_j - 2 \sum_{i=1}^M E_{J,i} |\cos(2\pi f_i)| \cos \hat{\varphi}_i$$



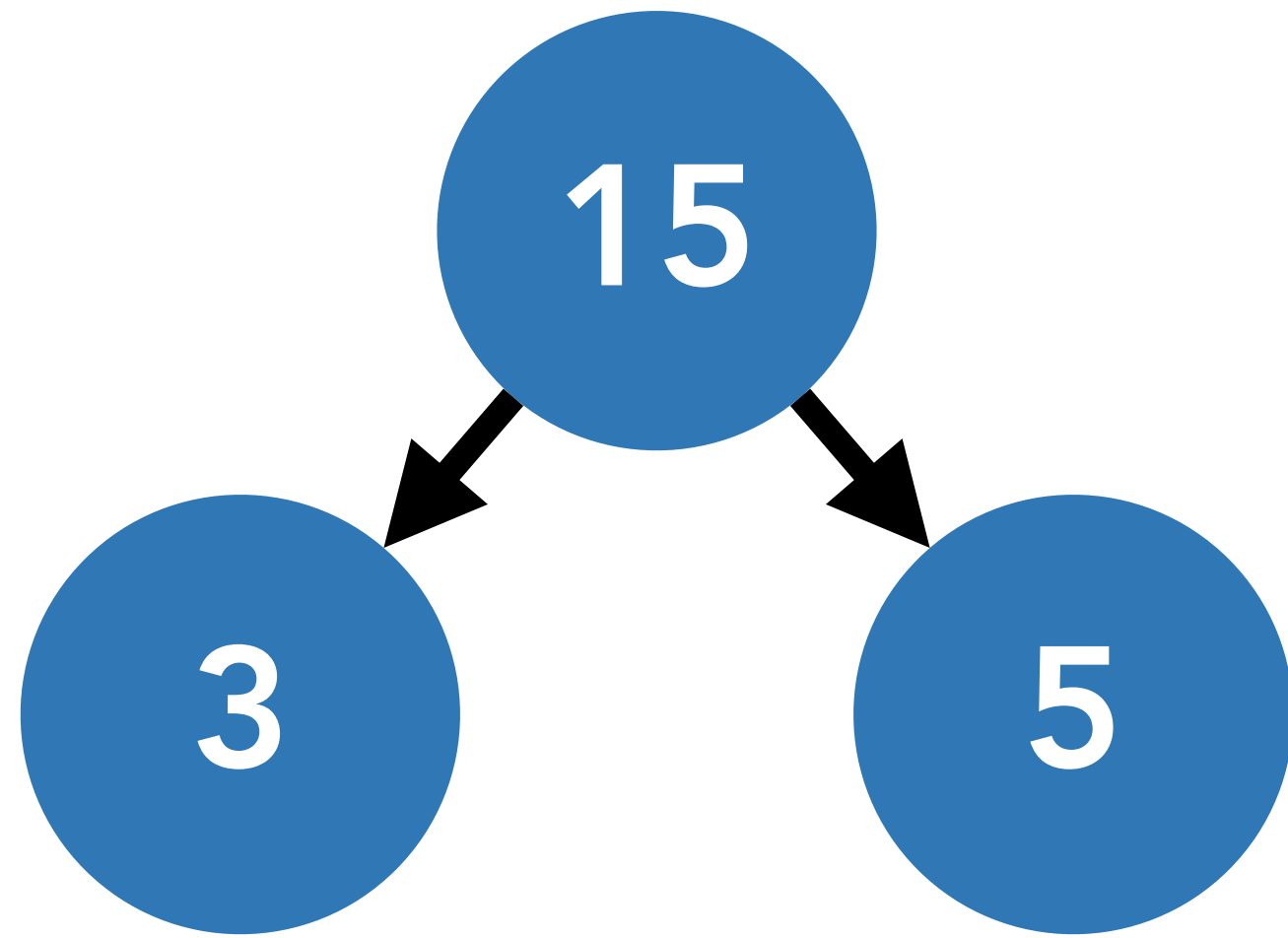
Advantage of QIPA over QITE for early steps over a broad parameter range

T. H. Kyaw*, [Micheline B. Soley](#)*, B. Allen, P. Bergold, C. Sun, V. S. Batista, A. Aspuru-Guzik, (2022) arXiv:2208.10470v1.

T. H. Kyaw*, Ti. Menke*, et al. (2021) arXiv:2006.03070v3.

QIPA FOR PRIME FACTORIZATION (CYBERSECURITY)

QIPA converges **faster than QITE** for all integers factored.

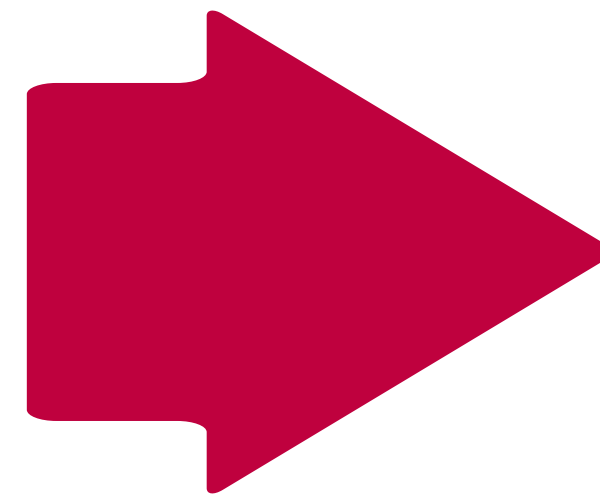


$$N = p \times q$$

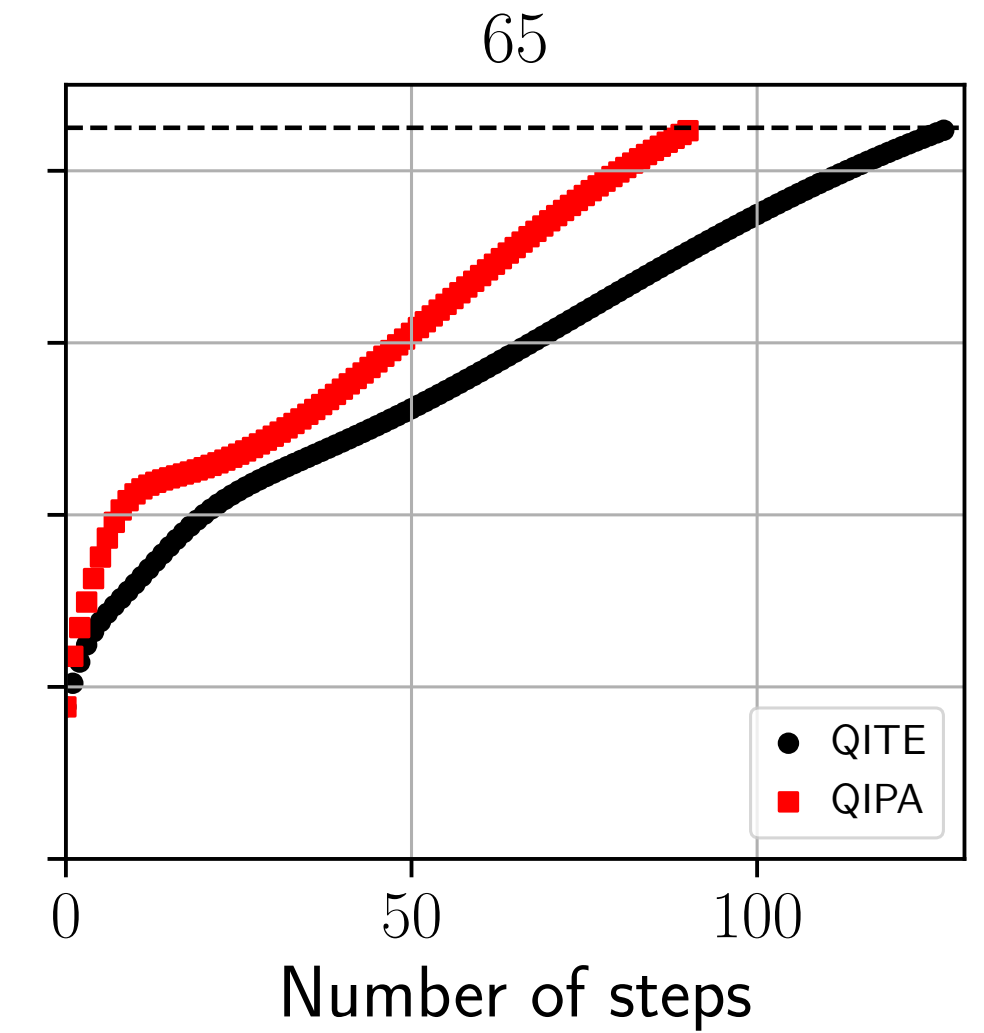
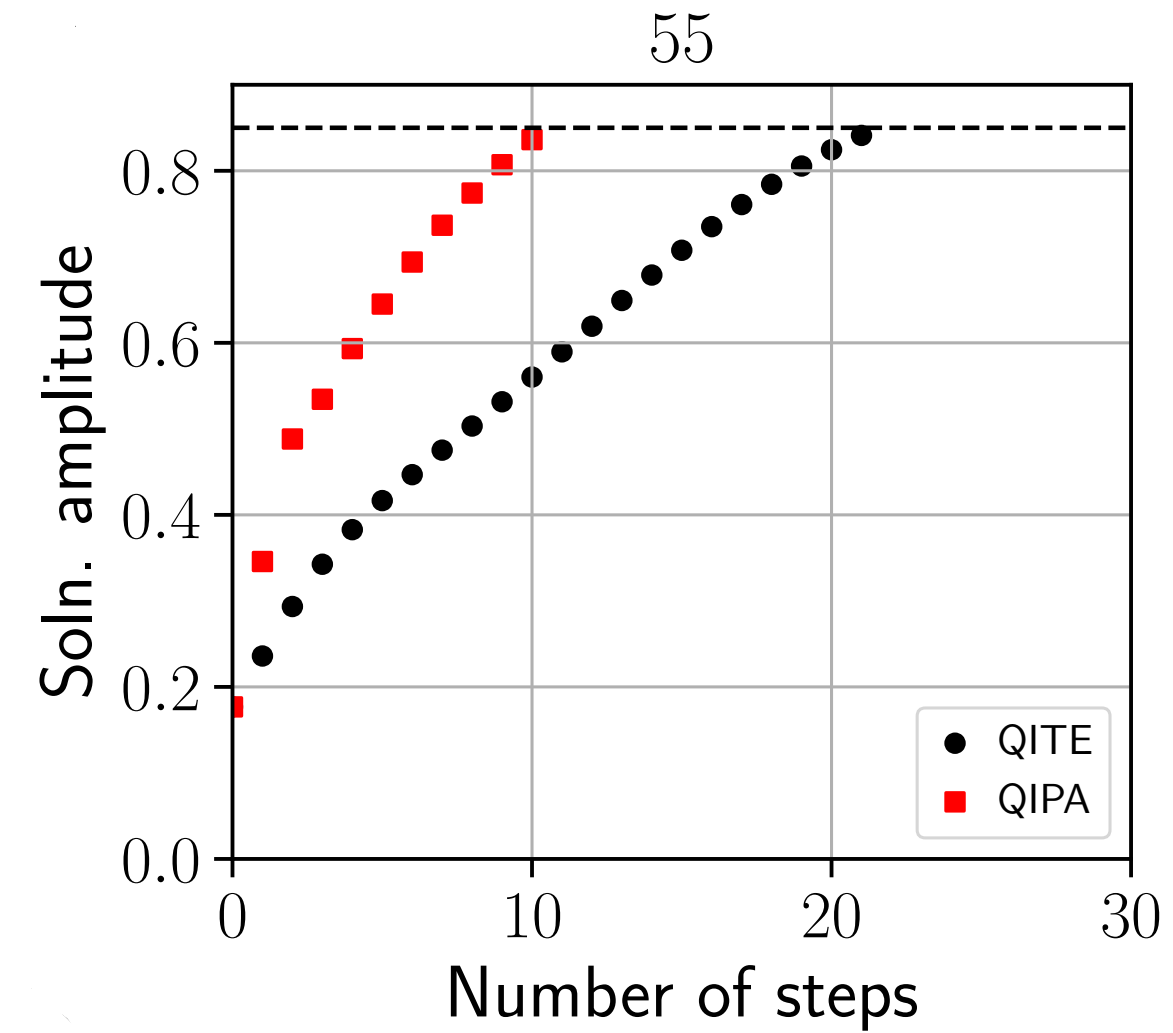
$$p = H_N(q, p) = d(N; q, p)^2, \quad d(N; q, p) = N - q \times p$$

$$d(N, \vec{x}) = N - \left(1 + \sum_{j=1}^L x_j 2^j \right) \times \left(1 + \sum_{k=1}^L x_{L+k} 2^k \right)$$

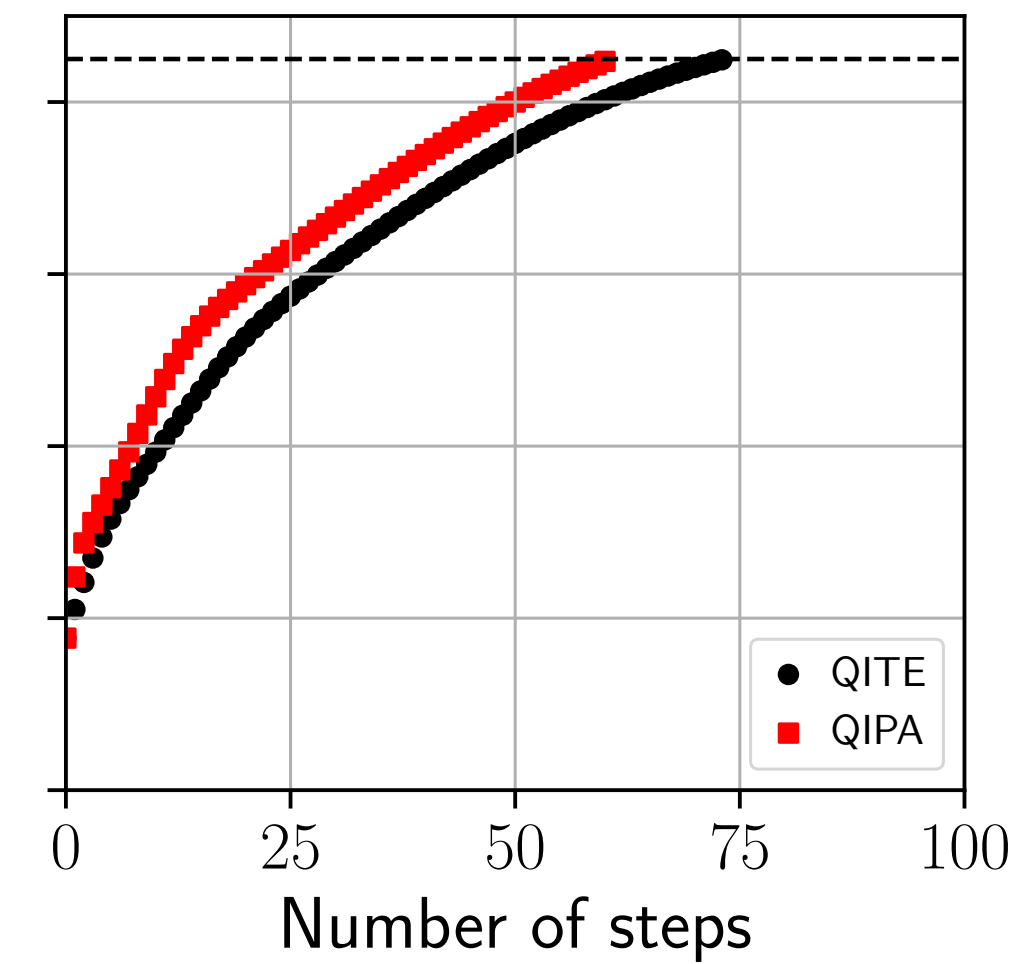
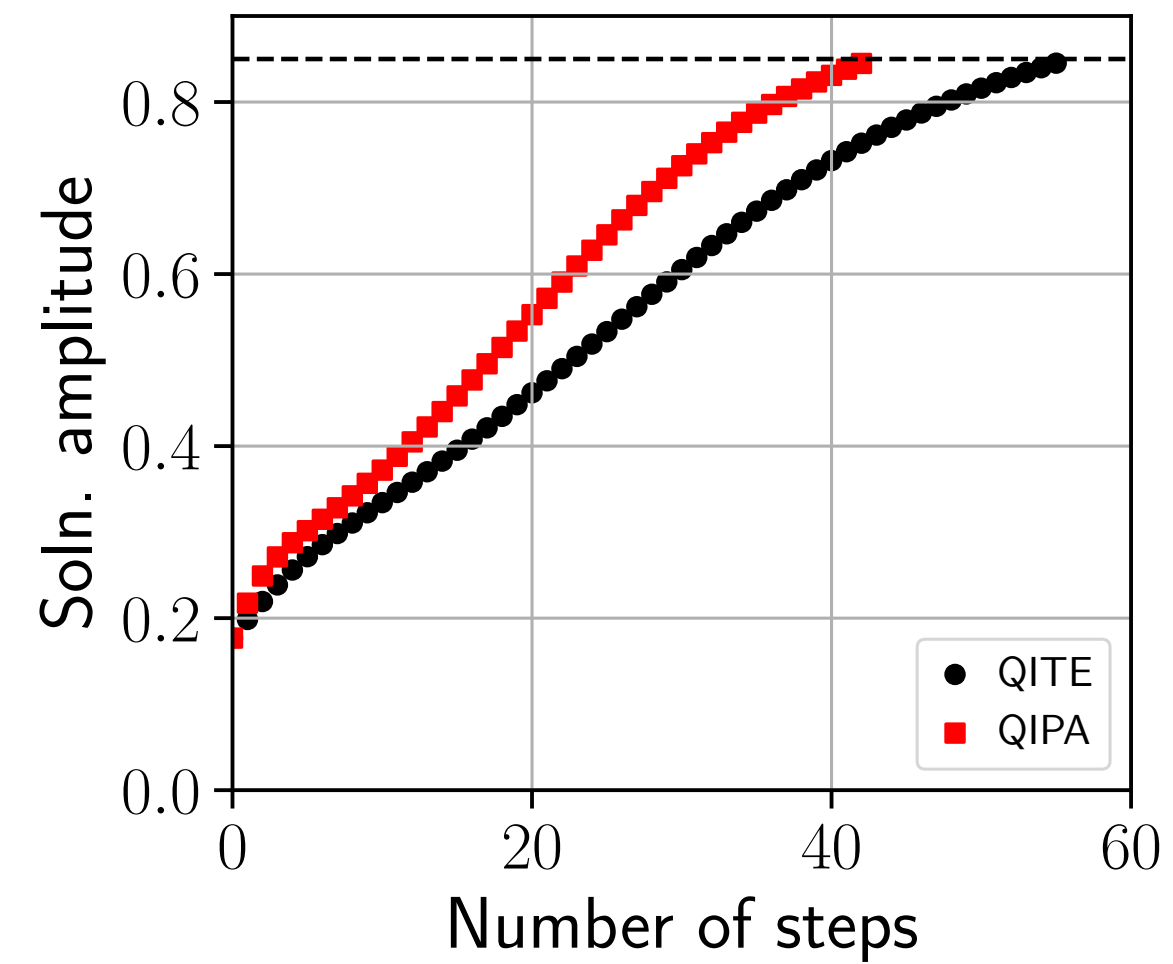
$$d(N; \vec{s}) = N - \left(2^L + \sum_{j=1}^L s_j 2^{j-1} \right) \times \left(2^L + \sum_{k=1}^L s_{L+k} 2^{k-1} \right)$$



YZ
Ansatz

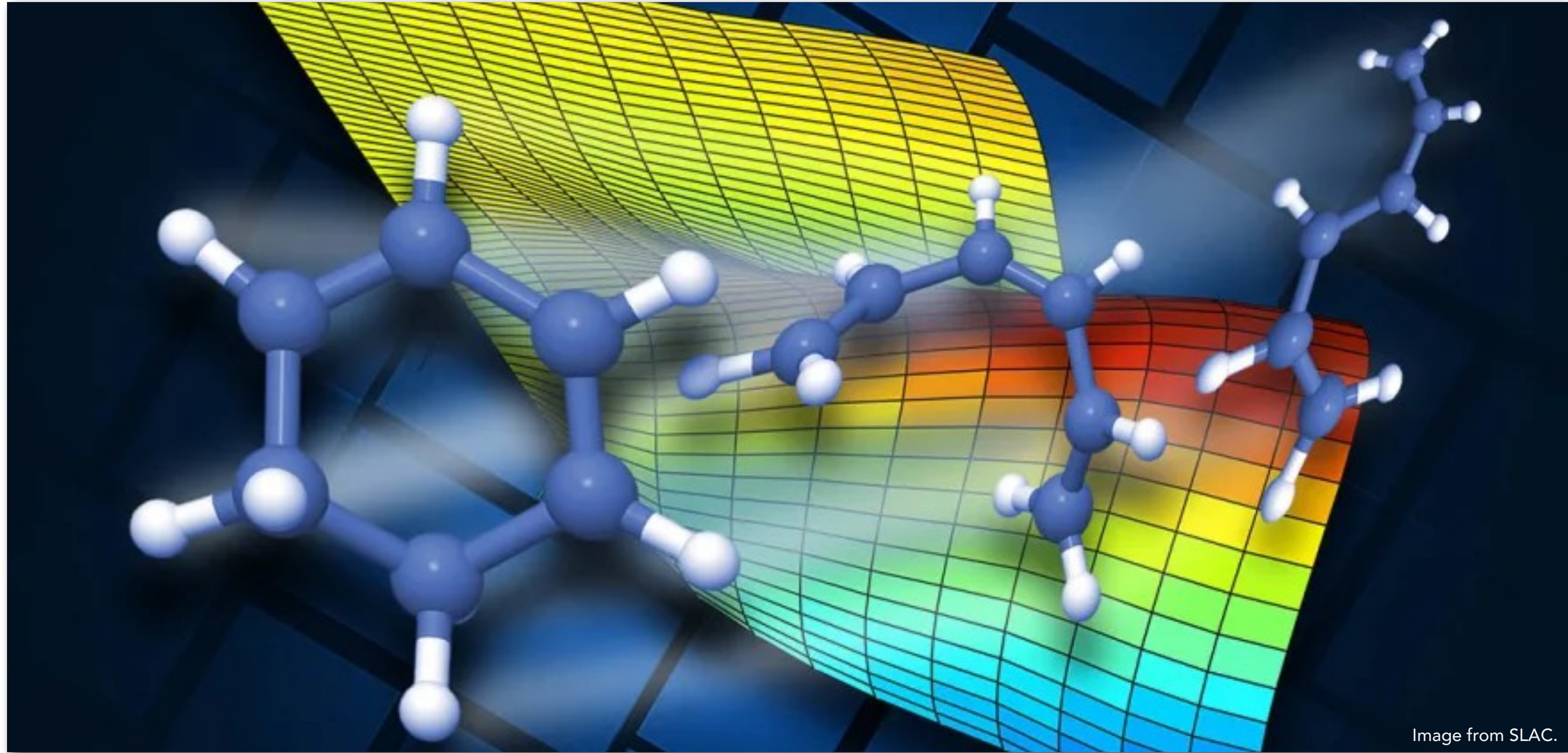


Y
Ansatz



OUTLOOK

IPA's efficiency and ability to avoid local minima to converge deterministically to global optima makes the method well-suited to a wide range of global optimization problems in chemistry beyond reach with standard grid-based methods.



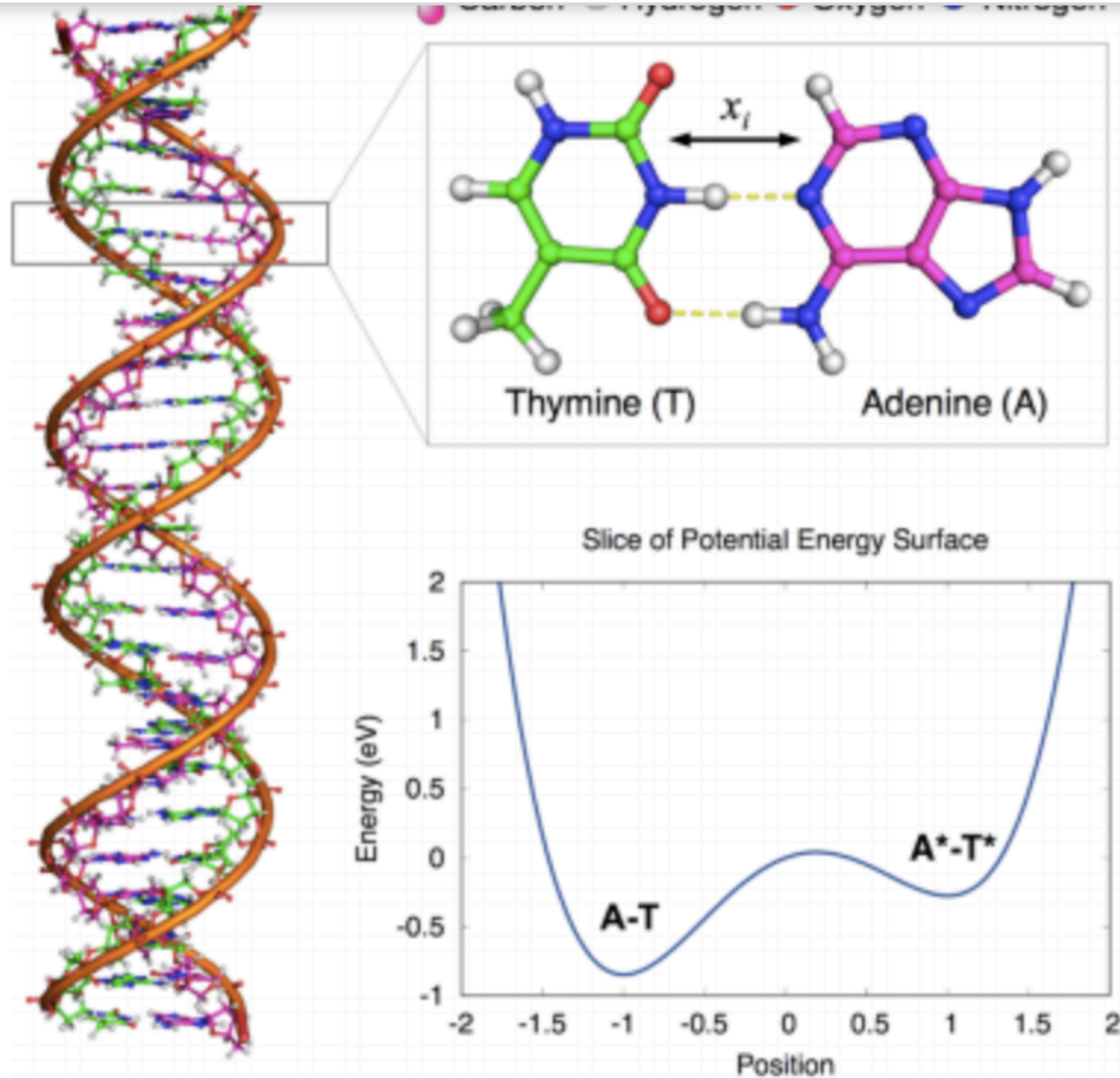
T. H. Kyaw*, **Micheline B. Soley***, B. Allen, P. Bergold, C. Sun, V. S. Batista, A. Aspuru-Guzik, (2022) arXiv:2208.10470v1.

Micheline B. Soley, P. Bergold, V. S. Batista, JCTC 17 (2021) 3280.



+ Code + Text

Connect



Above: DNA chain (left) of $D = 50$ hydrogen bonds corresponding to 25 hydrogen-bonded adenine–thymine base pairs (inset, top right), with hydrogen bonds shown as dashed yellow lines. Each hydrogen bonded proton attaches to either base, with energy represented by the double-well potential (bottom right).

```
[ ] beta=10
dim=2
ens=1.0e-14
```

ACKNOWLEDGEMENTS



University of Wisconsin-Madison Office of the
Vice Chancellor for Research and Graduate
Education

BLUE WATERS

Blue Waters Graduate Research Fellowship,
Supported by NSF (OCI-0725070, ACI-1238993)
and the State of Illinois, Joint Effort of UIUC
and NCSA



WARF

Wisconsin Alumni Research Foundation

Wisconsin Alumni Research Fund



NSF GRFP (DGE-1144152),
NSF Grant No. CHE-1900160,
NSF CCI Center for
Quantum Dynamics on Modular
Quantum Devices (2124511)



NERSC

YQI

YQI Postdoctoral Fellowship

Y | CRC

Yale High Performance
Computing Center



Harvard GSAS
Merit/Graduate Society
Fellowship