

quantumfddd, a computational framework for the Relativistic Schrödinger Equation

Rafael L. Delgado



POLITÉCNICA

UNIVERSIDAD
POLITÉCNICA
DE MADRID

Based on Comput. Phys. Commun. **272** (2022) 108250

Code on https://github.com/quantumfddd/quantumfddd_v3

XVth Quark Confinement and the Hadron Spectrum, Stavanger (Norway)

Introduction

- The three-dimensional Schrödinger equation has analytical solutions for only a very small class of systems.
- Most phenomenological descriptions of QCD bound states are described by the 3D Schrödinger equation with a wide variety of potentials.
- Examples: below-threshold charmonium production and bottomonium spectra; potential-based non-relativistic QCD (pNRQCD); quarkonium evolution in the quark-gluon plasma,...
- Goal: finding numerical solutions of the Three-Dimensional Schrödinger Equation with arbitrary potentials.
- Means: we extend the parallelized solver called `quantumfddtd` from M. Strickland et al. [J.Compt.Phys.**229**, 6015; PRD**83**, 105019].
- Code available under a GPLv3 license on https://github.com/quantumfddtd/quantumfddtd_v3

Introduction

- The three-dimensional Schrödinger equation has analytical solutions for only a very small class of systems.
- Most phenomenological descriptions of QCD bound states are described by the 3D Schrödinger equation with a wide variety of potentials.
- Examples: below-threshold charmonium production and bottomonium spectra; potential-based non-relativistic QCD (pNRQCD); quarkonium evolution in the quark-gluon plasma,...
- Goal: finding numerical solutions of the Three-Dimensional Schrödinger Equation with arbitrary potentials.
- Means: we extend the parallelized solver called `quantumfddtd` from M. Strickland et al. [J.Compt.Phys.**229**, 6015; PRD**83**, 105019].
- Code available under a GPLv3 license on https://github.com/quantumfddtd/quantumfddtd_v3

Introduction

- The three-dimensional Schrödinger equation has analytical solutions for only a very small class of systems.
- Most phenomenological descriptions of QCD bound states are described by the 3D Schrödinger equation with a wide variety of potentials.
- Examples: below-threshold charmonium production and bottomonium spectra; potential-based non-relativistic QCD (pNRQCD); quarkonium evolution in the quark-gluon plasma,...
- Goal: finding numerical solutions of the Three-Dimensional Schrödinger Equation with arbitrary potentials.
- Means: we extend the parallelized solver called `quantumfdd` from M. Strickland et al. [J.Compt.Phys.**229**, 6015; PRD**83**, 105019].
- Code available under a GPLv3 license on https://github.com/quantumfdd/quantumfdd_v3

Introduction

- The three-dimensional Schrödinger equation has analytical solutions for only a very small class of systems.
- Most phenomenological descriptions of QCD bound states are described by the 3D Schrödinger equation with a wide variety of potentials.
- Examples: below-threshold charmonium production and bottomonium spectra; potential-based non-relativistic QCD (pNRQCD); quarkonium evolution in the quark-gluon plasma,...
- Goal: finding numerical solutions of the Three-Dimensional Schrödinger Equation with arbitrary potentials.
- Means: we extend the parallelized solver called `quantumfddtd` from M. Strickland et al. [J.Compt.Phys.**229**, 6015; PRD**83**, 105019].
- Code available under a GPLv3 license on https://github.com/quantumfddtd/quantumfddtd_v3

Introduction

- The three-dimensional Schrödinger equation has analytical solutions for only a very small class of systems.
- Most phenomenological descriptions of QCD bound states are described by the 3D Schrödinger equation with a wide variety of potentials.
- Examples: below-threshold charmonium production and bottomonium spectra; potential-based non-relativistic QCD (pNRQCD); quarkonium evolution in the quark-gluon plasma,...
- Goal: finding numerical solutions of the Three-Dimensional Schrödinger Equation with arbitrary potentials.
- Means: we extend the parallelized solver called `quantumfdd` from M. Strickland et al. [J.Compt.Phys.**229**, 6015; PRD**83**, 105019].
- Code available under a GPLv3 license on https://github.com/quantumfdd/quantumfdd_v3

Introduction

- The three-dimensional Schrödinger equation has analytical solutions for only a very small class of systems.
- Most phenomenological descriptions of QCD bound states are described by the 3D Schrödinger equation with a wide variety of potentials.
- Examples: below-threshold charmonium production and bottomonium spectra; potential-based non-relativistic QCD (pNRQCD); quarkonium evolution in the quark-gluon plasma,...
- Goal: finding numerical solutions of the Three-Dimensional Schrödinger Equation with arbitrary potentials.
- Means: we extend the parallelized solver called `quantumfdd` from M. Strickland et al. [J.Compt.Phys.**229**, 6015; PRD**83**, 105019].
- Code available under a GPLv3 license on https://github.com/quantumfdd/quantumfdd_v3

The solver

- We implement full 3D kinetic terms, hence there is **no restriction over the symmetry** of the (arbitrary) external potential.
- `quantumfdd` is coded in C++ and uses the MPI, FFTW_MPI v.3, GNU Scientific Library (GSL), and CBLAS libraries.
- For the post-processing scripts, Python 3 (with Pandas library), gnuplot and Bash are employed.
- Input: the program accepts a configuration file (usually, on `input/params.txt` and (optional) command line arguments.
- New feature: the program can accept arbitrary external potentials as ASCII files, or you can use the hard-coded potentials. You could easily extend these hardcoded potentials to include your own ones as well.
- Output: ASCII files (usually on `data/` and `data/snapshot`).
- Post-processing scripts: Python scripts that allow spherical harmonics projections, symmetrization,...

The solver

- We implement full 3D kinetic terms, hence there is **no restriction over the symmetry** of the (arbitrary) external potential.
- `quantumfdtd` is coded in C++ and uses the MPI, FFTW_MPI v.3, GNU Scientific Library (GSL), and CBLAS libraries.
- For the post-processing scripts, Python 3 (with Pandas library), gnuplot and Bash are employed.
- Input: the program accepts a configuration file (usually, on `input/params.txt` and (optional) command line arguments.
- New feature: the program can accept arbitrary external potentials as ASCII files, or you can use the hard-coded potentials. You could easily extend these hardcoded potentials to include your own ones as well.
- Output: ASCII files (usually on `data/` and `data/snapshot`).
- Post-processing scripts: Python scripts that allow spherical harmonics projections, symmetrization,...

The solver

- We implement full 3D kinetic terms, hence there is **no restriction over the symmetry** of the (arbitrary) external potential.
- `quantumfdtd` is coded in C++ and uses the MPI, FFTW_MPI v.3, GNU Scientific Library (GSL), and CBLAS libraries.
- For the post-processing scripts, Python 3 (with Pandas library), gnuplot and Bash are employed.
- Input: the program accepts a configuration file (usually, on `input/params.txt` and (optional) command line arguments.
- New feature: the program can accept arbitrary external potentials as ASCII files, or you can use the hard-coded potentials. You could easily extend these hardcoded potentials to include your own ones as well.
- Output: ASCII files (usually on `data/` and `data/snapshot`).
- Post-processing scripts: Python scripts that allow spherical harmonics projections, symmetrization,...

The solver

- We implement full 3D kinetic terms, hence there is **no restriction over the symmetry** of the (arbitrary) external potential.
- `quantumfdtd` is coded in C++ and uses the MPI, FFTW_MPI v.3, GNU Scientific Library (GSL), and CBLAS libraries.
- For the post-processing scripts, Python 3 (with Pandas library), gnuplot and Bash are employed.
- Input: the program accepts a configuration file (usually, on `input/params.txt` and (optional) command line arguments.
- New feature: the program can accept arbitrary external potentials as ASCII files, or you can use the hard-coded potentials. You could easily extend these hardcoded potentials to include your own ones as well.
- Output: ASCII files (usually on `data/` and `data/snapshot`).
- Post-processing scripts: Python scripts that allow spherical harmonics projections, symmetrization,...

The solver

- We implement full 3D kinetic terms, hence there is **no restriction over the symmetry** of the (arbitrary) external potential.
- `quantumfdtd` is coded in C++ and uses the MPI, FFTW_MPI v.3, GNU Scientific Library (GSL), and CBLAS libraries.
- For the post-processing scripts, Python 3 (with Pandas library), gnuplot and Bash are employed.
- Input: the program accepts a configuration file (usually, on `input/params.txt` and (optional) command line arguments.
- New feature: the program can accept arbitrary external potentials as ASCII files, or you can use the hard-coded potentials. You could easily extend these hardcoded potentials to include your own ones as well.
- Output: ASCII files (usually on `data/` and `data/snapshot`).
- Post-processing scripts: Python scripts that allow spherical harmonics projections, symmetrization,...

The solver

- We implement full 3D kinetic terms, hence there is **no restriction over the symmetry** of the (arbitrary) external potential.
- `quantumfdtd` is coded in C++ and uses the MPI, FFTW_MPI v.3, GNU Scientific Library (GSL), and CBLAS libraries.
- For the post-processing scripts, Python 3 (with Pandas library), gnuplot and Bash are employed.
- Input: the program accepts a configuration file (usually, on `input/params.txt` and (optional) command line arguments.
- New feature: the program can accept arbitrary external potentials as ASCII files, or you can use the hard-coded potentials. You could easily extend these hardcoded potentials to include your own ones as well.
- Output: ASCII files (usually on `data/` and `data/snapshot`).
- Post-processing scripts: Python scripts that allow spherical harmonics projections, symmetrization,...

The solver

- We implement full 3D kinetic terms, hence there is **no restriction over the symmetry** of the (arbitrary) external potential.
- `quantumfdd` is coded in C++ and uses the MPI, FFTW_MPI v.3, GNU Scientific Library (GSL), and CBLAS libraries.
- For the post-processing scripts, Python 3 (with Pandas library), gnuplot and Bash are employed.
- Input: the program accepts a configuration file (usually, on `input/params.txt` and (optional) command line arguments.
- New feature: the program can accept arbitrary external potentials as ASCII files, or you can use the hard-coded potentials. You could easily extend these hardcoded potentials to include your own ones as well.
- Output: ASCII files (usually on `data/` and `data/snapshot`).
- Post-processing scripts: Python scripts that allow spherical harmonics projections, symmetrization,...

New features in quantumfdd: relativistic Schrödinger equation

- The Schrödinger Hamiltonian is split into a kinetic term, H_K , and a potential $V(\vec{r})$,

$$H = H_K + V(\vec{r})$$

- The usual non-relativistic term is $H_K^{nr} = \sum_{i=1,2,3} \frac{p_i^2}{2m}$, where $\vec{p} = (p_1, p_2, p_3)$ is the spatial three momentum.
- This non-relativistic term was coded in old quantumfdd.
- New in quantumfdd v3: relativistic kinetic term,

$$H_K^{rel} = \sqrt{m^2 + \sum_{i=1,2,3} p_i^2}$$

New features in quantumfdd: relativistic Schrödinger equation

- The Schrödinger Hamiltonian is split into a kinetic term, H_K , and a potential $V(\vec{r})$,

$$H = H_K + V(\vec{r})$$

- The usual non-relativistic term is $H_K^{nr} = \sum_{i=1,2,3} \frac{p_i^2}{2m}$, where $\vec{p} = (p_1, p_2, p_3)$ is the spatial three momentum.
- This non-relativistic term was coded in old quantumfdd.
- New in quantumfdd v3: relativistic kinetic term,

$$H_K^{rel} = \sqrt{m^2 + \sum_{i=1,2,3} p_i^2}$$

New features in quantumfdd: relativistic Schrödinger equation

- The Schrödinger Hamiltonian is split into a kinetic term, H_K , and a potential $V(\vec{r})$,

$$H = H_K + V(\vec{r})$$

- The usual non-relativistic term is $H_K^{nr} = \sum_{i=1,2,3} \frac{p_i^2}{2m}$, where $\vec{p} = (p_1, p_2, p_3)$ is the spatial three momentum.
- This non-relativistic term was coded in old quantumfdd.
- New in quantumfdd v3: relativistic kinetic term,

$$H_K^{rel} = \sqrt{m^2 + \sum_{i=1,2,3} p_i^2}$$

New features in quantumfdd: relativistic Schrödinger equation

- The Schrödinger Hamiltonian is split into a kinetic term, H_K , and a potential $V(\vec{r})$,

$$H = H_K + V(\vec{r})$$

- The usual non-relativistic term is $H_K^{nr} = \sum_{i=1,2,3} \frac{p_i^2}{2m}$, where $\vec{p} = (p_1, p_2, p_3)$ is the spatial three momentum.
- This non-relativistic term was coded in old quantumfdd.
- New in quantumfdd v3: relativistic kinetic term,

$$H_K^{rel} = \sqrt{m^2 + \sum_{i=1,2,3} p_i^2}$$

Evaluation of the kinetic term: FDTD

- Old quantumfddtd uses finite-difference time-domain (FDTD) method with **Dirichlet** boundary conditions, $\Psi(\text{boundary}) \equiv 0$:

$$H_K^{(0)}\Psi = -\frac{1}{2M} \sum_{l=\pm 1}^{\pm 3} \frac{1}{2A} \frac{\Psi(\vec{r} + \hat{e}_l) - \Psi(\vec{r})}{A},$$

$$\vec{r} \equiv (Ax_1, Ax_2, Ax_3), \quad x_1, x_2, x_3 \in \{0, 1, \dots, N-1\}$$

- On momentum space, the equivalent Hamiltonian is:

$$H_K^{(0)}\Psi(\vec{p}) = \frac{1}{2M} \sum_{l=1}^3 \frac{4}{A^2} \sin^2\left(\frac{Ap_l}{2}\right) \Psi(\vec{p}), \quad k_l \equiv Ap_l$$

Evaluation of the kinetic term: FDTD

- Old quantumfddtd uses finite-difference time-domain (FDTD) method with **Dirichlet** boundary conditions, $\Psi(\text{boundary}) \equiv 0$:

$$H_K^{(0)}\Psi = -\frac{1}{2M} \sum_{l=\pm 1}^{\pm 3} \frac{1}{2A} \frac{\Psi(\vec{r} + \hat{e}_l) - \Psi(\vec{r})}{A},$$

$$\vec{r} \equiv (Ax_1, Ax_2, Ax_3), \quad x_1, x_2, x_3 \in \{0, 1, \dots, N-1\}$$

- On momentum space, the equivalent Hamiltonian is:

$$H_K^{(0)}\Psi(\vec{p}) = \frac{1}{2M} \sum_{l=1}^3 \frac{4}{A^2} \sin^2\left(\frac{Ap_l}{2}\right) \Psi(\vec{p}), \quad k_l \equiv Ap_l$$

Fast Fourier Transform

- Three new kinetic terms ($H_K^{(1,2,3)}$): we use the Fast Fourier Transform (FFT) to go from position to momentum space; evaluate H_K ; and use the Inverse FFT (IFFT) to go back to position space.
- FFT \Rightarrow **periodic** boundary conditions.
- $H_K^{(1)}$ and $H_K^{(2)}$: non-relativistic kinetic terms; $H_K^{(1)}$ uses naive FFT-based differentiation; $H_K^{(2)}$ uses the correct momentum space (via Symanzik effective field theory) on a finite and periodic lattice.

$$H_K^{(1)}\Psi = \frac{1}{2A^2MN^3} \cdot \text{IFFT} \left[\sum_{l=1}^3 (k_l)^2 \cdot \text{FFT}[\Psi] \right]$$

$$H_K^{(2)}\Psi = \frac{1}{2A^2MN^3} \cdot \text{IFFT} \left[4 \sum_{l=1}^3 \sin^2\left(\frac{k_l}{2}\right) \cdot \text{FFT}[\Psi] \right]$$

Fast Fourier Transform

- Three new kinetic terms ($H_K^{(1,2,3)}$): we use the Fast Fourier Transform (FFT) to go from position to momentum space; evaluate H_K ; and use the Inverse FFT (IFFT) to go back to position space.
- FFT \implies **periodic** boundary conditions.
- $H_K^{(1)}$ and $H_K^{(2)}$: non-relativistic kinetic terms; $H_K^{(1)}$ uses naive FFT-based differentiation; $H_K^{(2)}$ uses the correct momentum space (via Symanzik effective field theory) on a finite and periodic lattice.

$$H_K^{(1)}\Psi = \frac{1}{2A^2MN^3} \cdot \text{IFFT} \left[\sum_{l=1}^3 (k_l)^2 \cdot \text{FFT}[\Psi] \right]$$

$$H_K^{(2)}\Psi = \frac{1}{2A^2MN^3} \cdot \text{IFFT} \left[4 \sum_{l=1}^3 \sin^2\left(\frac{k_l}{2}\right) \cdot \text{FFT}[\Psi] \right]$$

Fast Fourier Transform

- Three new kinetic terms ($H_K^{(1,2,3)}$): we use the Fast Fourier Transform (FFT) to go from position to momentum space; evaluate H_K ; and use the Inverse FFT (IFFT) to go back to position space.
- FFT \implies **periodic** boundary conditions.
- $H_K^{(1)}$ and $H_K^{(2)}$: non-relativistic kinetic terms; $H_K^{(1)}$ uses naive FFT-based differentiation; $H_K^{(2)}$ uses the correct momentum space (via Symanzik effective field theory) on a finite and periodic lattice.

$$H_K^{(1)}\psi = \frac{1}{2A^2MN^3} \cdot \text{IFFT} \left[\sum_{l=1}^3 (k_l)^2 \cdot \text{FFT}[\psi] \right]$$

$$H_K^{(2)}\psi = \frac{1}{2A^2MN^3} \cdot \text{IFFT} \left[4 \sum_{l=1}^3 \sin^2\left(\frac{k_l}{2}\right) \cdot \text{FFT}[\psi] \right]$$

The iterative procedure

- In order to extract the ground, second and third excited states wavefunctions ($\Psi_{0,1,2}$) and energies ($E_{0,1,2}$), we use the same iterative procedure that was coded on legacy quantumfdd, based on a Wick rotation of the Schrödinger Equation ($it \rightarrow \tau$)

$$\Psi(x_1, x_2, x_3, \tau + \Delta\tau) = \mathcal{A}\Psi(x_1, x_2, x_3, \tau) - \mathcal{B}\Delta\tau H_K \Psi(x_1, x_2, x_3, \tau)$$

$$\mathcal{A} = \frac{1 - \frac{\Delta\tau}{2} V(\vec{r})}{1 + \frac{\Delta\tau}{2} V(\vec{r})}, \quad \mathcal{B} = \frac{1}{1 + \frac{\Delta\tau}{2} V(\vec{r})}$$

- The evolution with τ is given by

$$\Psi(x_1, x_2, x_3, \tau) = \sum_{i=0}^{\infty} a_i \Psi_i(x_1, x_2, x_3) e^{-E_i \tau},$$

where $\{a_0, a_1, \dots\}$ are the decomposition coefficients of the initial guess $\Psi(x_1, x_2, x_3, 0)$ in the basis of eigenvectors.

- An overlap procedure is used for the numerical extraction of the first and second excited states.

The iterative procedure

- In order to extract the ground, second and third excited states wavefunctions ($\Psi_{0,1,2}$) and energies ($E_{0,1,2}$), we use the same iterative procedure that was coded on legacy quantumfdd, based on a Wick rotation of the Schrödinger Equation ($it \rightarrow \tau$)

$$\Psi(x_1, x_2, x_3, \tau + \Delta\tau) = \mathcal{A}\Psi(x_1, x_2, x_3, \tau) - \mathcal{B}\Delta\tau H_K \Psi(x_1, x_2, x_3, \tau)$$

$$\mathcal{A} = \frac{1 - \frac{\Delta\tau}{2} V(\vec{r})}{1 + \frac{\Delta\tau}{2} V(\vec{r})}, \quad \mathcal{B} = \frac{1}{1 + \frac{\Delta\tau}{2} V(\vec{r})}$$

- The evolution with τ is given by

$$\Psi(x_1, x_2, x_3, \tau) = \sum_{i=0}^{\infty} a_i \Psi_i(x_1, x_2, x_3) e^{-E_i \tau},$$

where $\{a_0, a_1, \dots\}$ are the decomposition coefficients of the initial guess $\Psi(x_1, x_2, x_3, 0)$ in the basis of eigenvectors.

- An overlap procedure is used for the numerical extraction of the first and second excited states.

The iterative procedure

- In order to extract the ground, second and third excited states wavefunctions ($\Psi_{0,1,2}$) and energies ($E_{0,1,2}$), we use the same iterative procedure that was coded on legacy quantumfdd, based on a Wick rotation of the Schrödinger Equation ($it \rightarrow \tau$)

$$\Psi(x_1, x_2, x_3, \tau + \Delta\tau) = \mathcal{A}\Psi(x_1, x_2, x_3, \tau) - \mathcal{B}\Delta\tau H_K \Psi(x_1, x_2, x_3, \tau)$$

$$\mathcal{A} = \frac{1 - \frac{\Delta\tau}{2} V(\vec{r})}{1 + \frac{\Delta\tau}{2} V(\vec{r})}, \quad \mathcal{B} = \frac{1}{1 + \frac{\Delta\tau}{2} V(\vec{r})}$$

- The evolution with τ is given by

$$\Psi(x_1, x_2, x_3, \tau) = \sum_{i=0}^{\infty} a_i \Psi_i(x_1, x_2, x_3) e^{-E_i \tau},$$

where $\{a_0, a_1, \dots\}$ are the decomposition coefficients of the initial guess $\Psi(x_1, x_2, x_3, 0)$ in the basis of eigenvectors.

- An overlap procedure is used for the numerical extraction of the first and second excited states.

Parity projections

- In order to extract the excited states, we include a Python script that allows for computing the positive P^+ , negative P^- and negative-around-an-axis $P_{\vec{p}_k}^-$ parity projections of a wave-function,

$$P^\pm \Psi(\vec{r}) = \frac{1}{2} [\Psi(\vec{r}) \pm \Psi(-\vec{r})]$$

$$P_{\vec{p}_k=\hat{e}_3}^- \Psi(x_1, x_2, x_3) = \frac{1}{2} [\Psi(x_1, x_2, x_3) + \Psi(-x_1, -x_2, x_3) \\ - \Psi(x_1, x_2, -x_3) - \Psi(-x_1, -x_2, -x_3)]$$

- We also include a script for enforcing the normalization of wave-functions within the lattice volume V , $\int_V dV |\Psi|^2 = 1$.
- These scripts are included on top of an option in `quantumfddtd` that allows for enforcing symmetry restrictions within the iterative procedure itself.

Parity projections

- In order to extract the excited states, we include a Python script that allows for computing the positive P^+ , negative P^- and negative-around-an-axis $P_{\vec{p}_k}^-$ parity projections of a wave-function,

$$P^\pm \Psi(\vec{r}) = \frac{1}{2} [\Psi(\vec{r}) \pm \Psi(-\vec{r})]$$

$$P_{\vec{p}_k=\hat{e}_3}^- \Psi(x_1, x_2, x_3) = \frac{1}{2} [\Psi(x_1, x_2, x_3) + \Psi(-x_1, -x_2, x_3) \\ - \Psi(x_1, x_2, -x_3) - \Psi(-x_1, -x_2, -x_3)]$$

- We also include a script for enforcing the normalization of wave-functions within the lattice volume V , $\int_V dV |\Psi|^2 = 1$.
- These scripts are included on top of an option in `quantumfddtd` that allows for enforcing symmetry restrictions within the iterative procedure itself.

Parity projections

- In order to extract the excited states, we include a Python script that allows for computing the positive P^+ , negative P^- and negative-around-an-axis $P_{\vec{p}_k}^-$ parity projections of a wave-function,

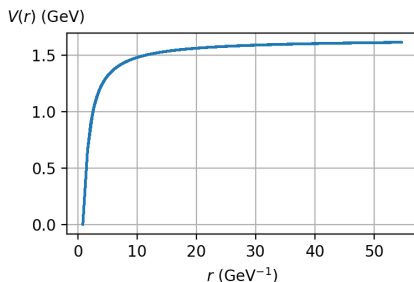
$$P^\pm \Psi(\vec{r}) = \frac{1}{2} [\Psi(\vec{r}) \pm \Psi(-\vec{r})]$$

$$P_{\vec{p}_k=\hat{e}_3}^- \Psi(x_1, x_2, x_3) = \frac{1}{2} [\Psi(x_1, x_2, x_3) + \Psi(-x_1, -x_2, x_3) \\ - \Psi(x_1, x_2, -x_3) - \Psi(-x_1, -x_2, -x_3)]$$

- We also include a script for enforcing the normalization of wave-functions within the lattice volume V , $\int_V dV |\Psi|^2 = 1$.
- These scripts are included on top of an option in `quantumfddtd` that allows for enforcing symmetry restrictions within the iterative procedure itself.

Initial conditions and studied cases

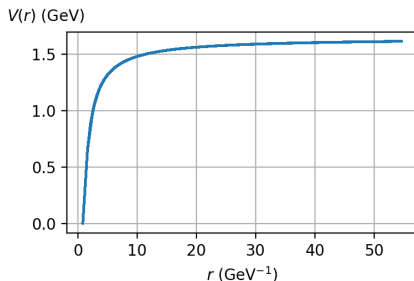
- We use a Coulomb potential for the following analysis.
- Initial guess: Coulomb-like initial conditions centered on the lattice volume. Actually, this is an initial guess with a linear combination of the states $1s$, $2s$, $2p$ ($m = 0$), and the real part of the $2p$ ($m = \pm 1$).



- Lattice spacing $A = 0.12 \text{ fm}$
- Unless otherwise stated, $N = 64$ centered on the lattice volume.
- Unless otherwise stated, $M = 0.3 \text{ GeV}$.
- $(A \cdot M)^2 \approx 0.03 \Rightarrow$ mild finite mass discretization errors

Initial conditions and studied cases

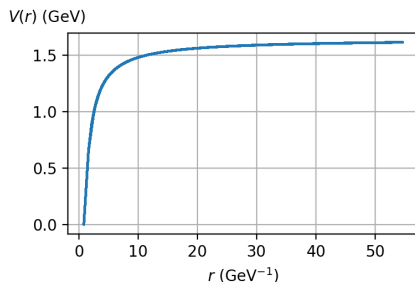
- We use a Coulomb potential for the following analysis.
- Initial guess: Coulomb-like initial conditions centered on the lattice volume. Actually, this is an initial guess with a linear combination of the states $1s$, $2s$, $2p$ ($m = 0$), and the real part of the $2p$ ($m = \pm 1$).



- Lattice spacing $A = 0.12 \text{ fm}$
- Unless otherwise stated, $N = 64$ centered on the lattice volume.
- Unless otherwise stated, $M = 0.3 \text{ GeV}$.
- $(A \cdot M)^2 \approx 0.03 \Rightarrow$ mild finite mass discretization errors

Initial conditions and studied cases

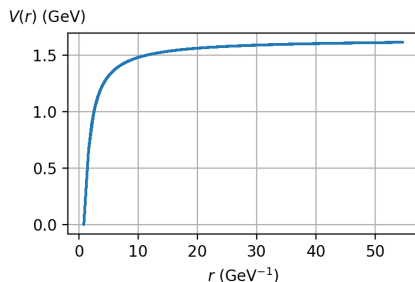
- We use a Coulomb potential for the following analysis.
- Initial guess: Coulomb-like initial conditions centered on the lattice volume. Actually, this is an initial guess with a linear combination of the states $1s$, $2s$, $2p$ ($m = 0$), and the real part of the $2p$ ($m = \pm 1$).



- Lattice spacing $A = 0.12$ fm
- Unless otherwise stated, $N = 64$ centered on the lattice volume.
- Unless otherwise stated, $M = 0.3$ GeV.
- $(A \cdot M)^2 \approx 0.03 \Rightarrow$ mild finite mass discretization errors

Initial conditions and studied cases

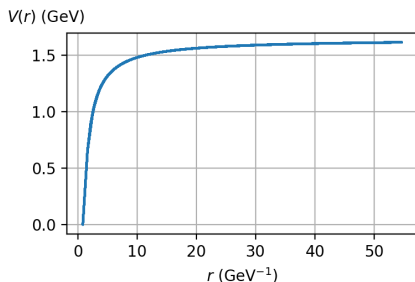
- We use a Coulomb potential for the following analysis.
- Initial guess: Coulomb-like initial conditions centered on the lattice volume. Actually, this is an initial guess with a linear combination of the states $1s$, $2s$, $2p$ ($m = 0$), and the real part of the $2p$ ($m = \pm 1$).



- Lattice spacing $A = 0.12 \text{ fm}$
- Unless otherwise stated, $N = 64$ centered on the lattice volume.
- Unless otherwise stated, $M = 0.3 \text{ GeV}$.
- $(A \cdot M)^2 \approx 0.03 \Rightarrow$ mild finite mass discretization errors

Initial conditions and studied cases

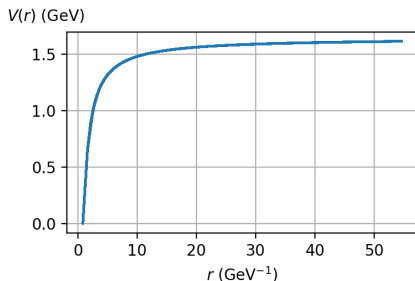
- We use a Coulomb potential for the following analysis.
- Initial guess: Coulomb-like initial conditions centered on the lattice volume. Actually, this is an initial guess with a linear combination of the states $1s$, $2s$, $2p$ ($m = 0$), and the real part of the $2p$ ($m = \pm 1$).



- Lattice spacing $A = 0.12 \text{ fm}$
- Unless otherwise stated, $N = 64$ centered on the lattice volume.
- Unless otherwise stated, $M = 0.3 \text{ GeV}$.
- $(A \cdot M)^2 \approx 0.03 \Rightarrow$ mild finite mass discretization errors

Initial conditions and studied cases

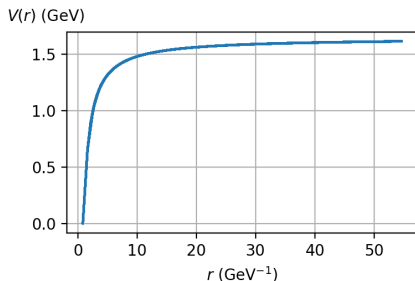
- We use a Coulomb potential for the following analysis.
- Initial guess: Coulomb-like initial conditions centered on the lattice volume. Actually, this is an initial guess with a linear combination of the states $1s$, $2s$, $2p$ ($m = 0$), and the real part of the $2p$ ($m = \pm 1$).



- Lattice spacing $A = 0.12 \text{ fm}$
- Unless otherwise stated, $N = 64$ centered on the lattice volume.
- Unless otherwise stated, $M = 0.3 \text{ GeV}$.
- $(A \cdot M)^2 \approx 0.03 \Rightarrow$ mild finite mass discretization errors

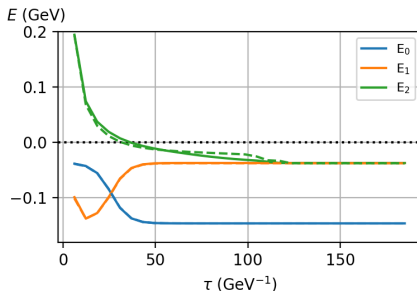
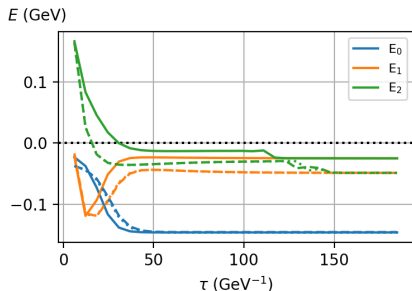
Initial conditions and studied cases

- We use a Coulomb potential for the following analysis.
- Initial guess: Coulomb-like initial conditions centered on the lattice volume. Actually, this is an initial guess with a linear combination of the states $1s$, $2s$, $2p$ ($m = 0$), and the real part of the $2p$ ($m = \pm 1$).



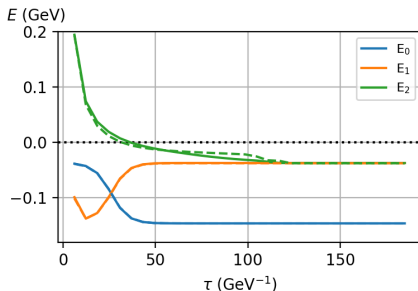
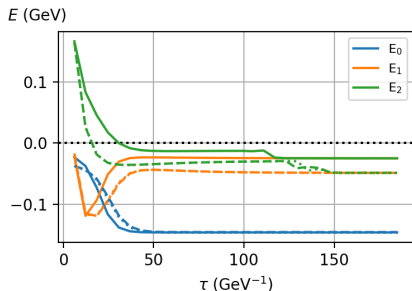
- Lattice spacing $A = 0.12 \text{ fm}$
- Unless otherwise stated, $N = 64$ centered on the lattice volume.
- Unless otherwise stated, $M = 0.3 \text{ GeV}$.
- $(A \cdot M)^2 \approx 0.03 \Rightarrow$ mild finite mass discretization errors

Comparison: different non-relativistic kinetic terms



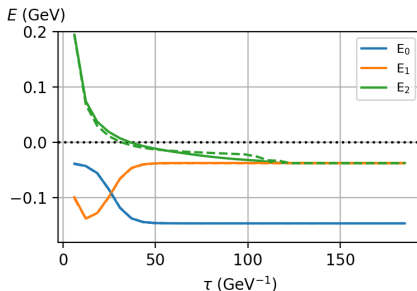
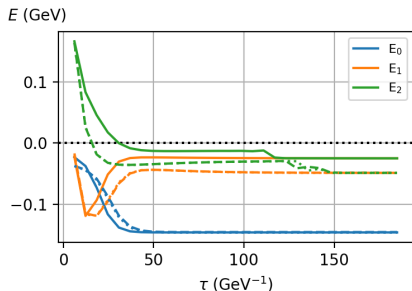
- $H_K^{(0)}$ (solid), $H_K^{(1)}$ (dashed) and $H_K^{(2)}$ (dotted). Left: $N = 64$; right: $N = 128$.
- Very small differences between $H_K^{(1)}$ and $H_K^{(2)}$, even with $N = 64$.
- The differences between $H_K^{(0)}$ (FDTD) and $H_K^{(2)}$ (FFT) kinetic terms that were evident with $N = 64$ are almost negligible with $N = 128$.
- $N = 64$: finite volume effects (boundary conditions!!) are evident. FDTD uses Dirichlet boundary cond. and FFT, periodic ones. Next slide.

Comparison: different non-relativistic kinetic terms



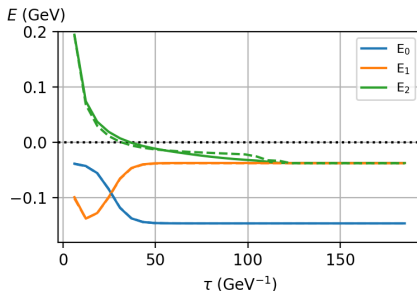
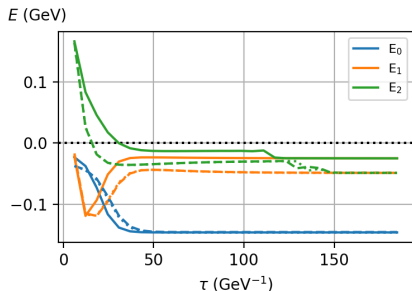
- $H_K^{(0)}$ (solid), $H_K^{(1)}$ (dashed) and $H_K^{(2)}$ (dotted). Left: $N = 64$; right: $N = 128$.
- Very small differences between $H_K^{(1)}$ and $H_K^{(2)}$, even with $N = 64$.
- The differences between $H_K^{(0)}$ (FDTD) and $H_K^{(2)}$ (FFT) kinetic terms that were evident with $N = 64$ are almost negligible with $N = 128$.
- $N = 64$: finite volume effects (boundary conditions!!) are evident. FDTD uses Dirichlet boundary cond. and FFT, periodic ones. Next slide.

Comparison: different non-relativistic kinetic terms



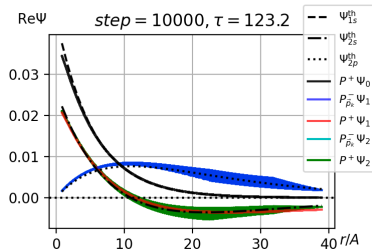
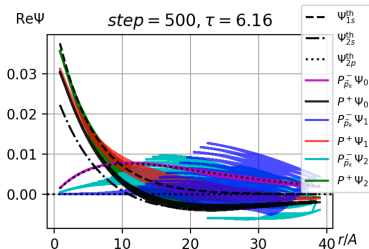
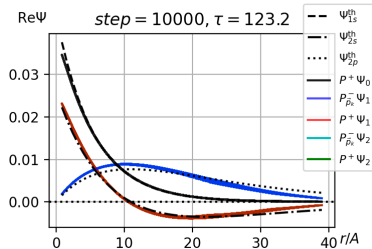
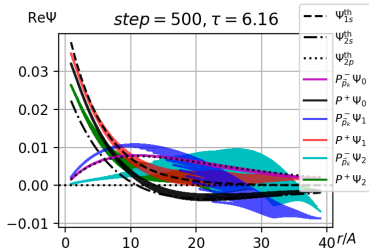
- $H_K^{(0)}$ (solid), $H_K^{(1)}$ (dashed) and $H_K^{(2)}$ (dotted). Left: $N = 64$; right: $N = 128$.
- Very small differences between $H_K^{(1)}$ and $H_K^{(2)}$, even with $N = 64$.
- The differences between $H_K^{(0)}$ (FDTD) and $H_K^{(2)}$ (FFT) kinetic terms that were evident with $N = 64$ are almost negligible with $N = 128$.
- $N = 64$: finite volume effects (boundary conditions!!) are evident. FDTD uses Dirichlet boundary cond. and FFT, periodic ones. Next slide.

Comparison: different non-relativistic kinetic terms



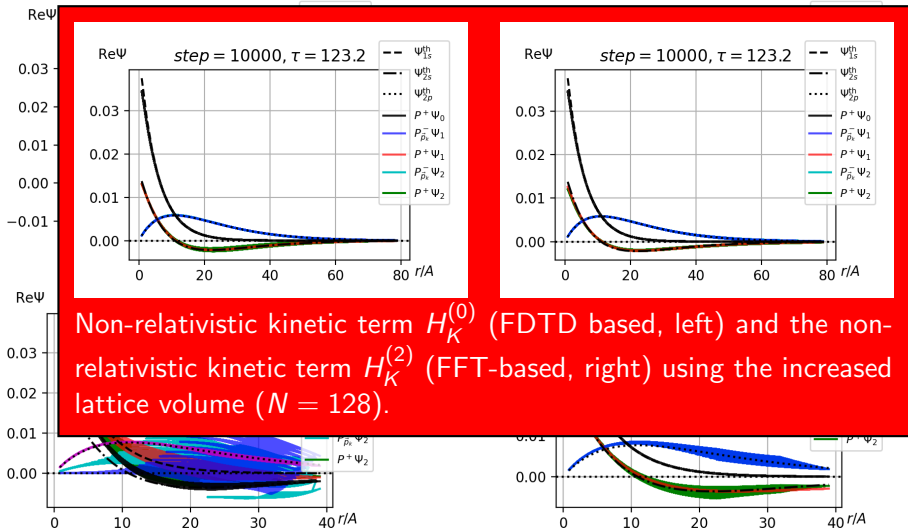
- $H_K^{(0)}$ (solid), $H_K^{(1)}$ (dashed) and $H_K^{(2)}$ (dotted). Left: $N = 64$; right: $N = 128$.
- Very small differences between $H_K^{(1)}$ and $H_K^{(2)}$, even with $N = 64$.
- The differences between $H_K^{(0)}$ (FDTD) and $H_K^{(2)}$ (FFT) kinetic terms that were evident with $N = 64$ are almost negligible with $N = 128$.
- $N = 64$: finite volume effects (boundary conditions!!) are evident. FDTD uses Dirichlet boundary cond. and FFT, periodic ones. Next slide.

Wave-functions for the non-relativistic $H_K^{(0)}$ and $H_K^{(2)}$



Up: $H_K^{(0)}$ (FDTD); down: $H_K^{(2)}$ (FFT)

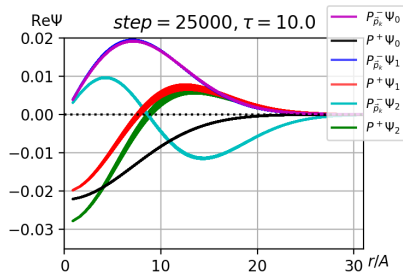
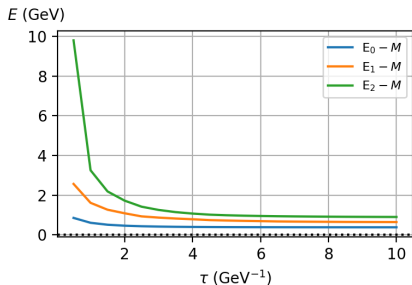
Wave-functions for the non-relativistic $H_K^{(0)}$ and $H_K^{(2)}$



Non-relativistic kinetic term $H_K^{(0)}$ (FDTD based, left) and the non-relativistic kinetic term $H_K^{(2)}$ (FFT-based, right) using the increased lattice volume ($N = 128$).

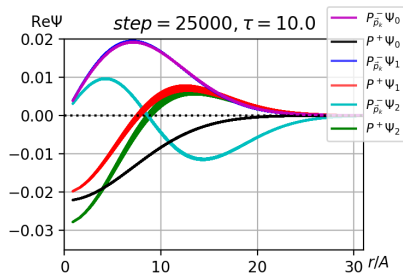
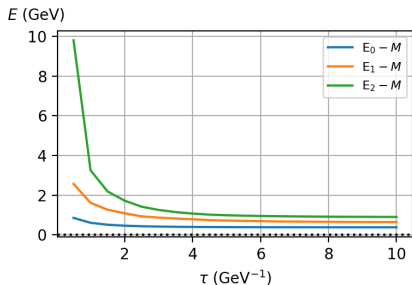
Up: $H_K^{(0)}$ (FDTD); down: $H_K^{(2)}$ (FFT)

Modified Harmonic oscillator: a test for the relativistic $H_K^{(3)}$



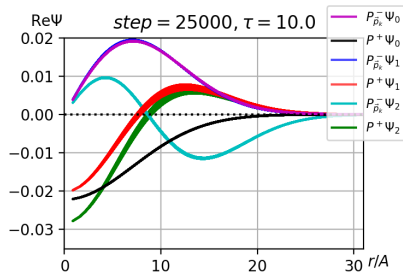
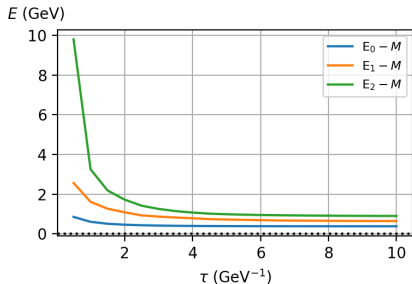
- In order to compare the $H_K^{(3)}$ kinetic term implementation with the literature, we are using the Harmonic oscillator, for which an analytical solution in momentum space exists [Z.F.Li et. al., J.Math.Phys.**46**, 103514].
- We have implemented such an analytical solution on Fortran:
https://github.com/quantumfdd/relativistic_harmonic_oscillator
- Appropriate parameters for comparison with the literature: $N = 256$, $M = 30 \text{ GeV}$, $A = 0.006 \text{ fm}$, $(A \cdot M)^2 \approx 0.81$ (discretization errors and large mass!).

Modified Harmonic oscillator: a test for the relativistic $H_K^{(3)}$



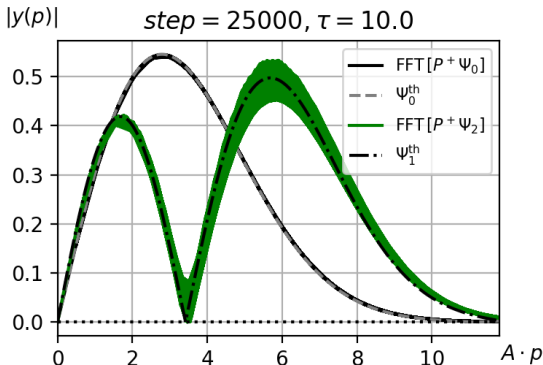
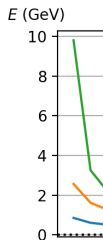
- In order to compare the $H_K^{(3)}$ kinetic term implementation with the literature, we are using the Harmonic oscillator, for which an analytical solution in momentum space exists [Z.F.Li et. al., J.Math.Phys.**46**, 103514].
- We have implemented such an analytical solution on Fortran:
https://github.com/quantumfdd/relativistic_harmonic_oscillator
- Appropriate parameters for comparison with the literature: $N = 256$, $M = 30 \text{ GeV}$, $A = 0.006 \text{ fm}$, $(A \cdot M)^2 \approx 0.81$ (discretization errors and large mass!).

Modified Harmonic oscillator: a test for the relativistic $H_K^{(3)}$



- In order to compare the $H_K^{(3)}$ kinetic term implementation with the literature, we are using the Harmonic oscillator, for which an analytical solution in momentum space exists [Z.F.Li et. al., J.Math.Phys.**46**, 103514].
- We have implemented such an analytical solution on Fortran:
https://github.com/quantumfdd/relativistic_harmonic_oscillator
- Appropriate parameters for comparison with the literature: $N = 256$, $M = 30 \text{ GeV}$, $A = 0.006 \text{ fm}$, $(A \cdot M)^2 \approx 0.81$ (discretization errors and large mass!).

Modified Harmonic oscillator: a test for the relativistic $H_K^{(3)}$



Momentum space. Solid, quantumfddd v.3 output turned into momentum space by means of the FFT; dashed, theoretical prediction of Ref. [J.Math.Phys.46, 103514].

- In order to compare the numerical results with the literature, we have solved the modified harmonic oscillator problem.
- We have used the parameters $M = 30 \text{ GeV}$, $A = 0.006 \text{ fm}$, $(A \cdot M)^2 \approx 0.81$ (discretization errors and large mass!).

Conclusions: extensions to quantumfdd

- We have extended the previous code `quantumfdd v.2` to `quantumfdd v.3`.
- We have implemented 2 additional non-relativistic kinetic terms, that are based on Fast Fourier Transform.
- We have implemented a new FFT-based relativistic kinetic term.
- The new code accepts arbitrary potential via external files according to a format described on [Comput.Phys.Commun.**272**, 108250]
- We allow for dumping full wave-functions as snapshots. This is specially valuable for extracting excited states.
- Small additional changes: cleanup of the code; we do not require an additional MPI node for controlling,...

Conclusions: extensions to quantumfddd

- We have extended the previous code quantumfddd v.2 to quantumfddd v.3.
- We have implemented 2 additional non-relativistic kinetic terms, that are based on Fast Fourier Transform.
- We have implemented a new FFT-based relativistic kinetic term.
- The new code accepts arbitrary potential via external files according to a format described on [Comput.Phys.Commun.**272**, 108250]
- We allow for dumping full wave-functions as snapshots. This is specially valuable for extracting excited states.
- Small additional changes: cleanup of the code; we do not require an additional MPI node for controlling,...

Conclusions: extensions to quantumfdd

- We have extended the previous code quantumfdd v.2 to quantumfdd v.3.
- We have implemented 2 additional non-relativistic kinetic terms, that are based on Fast Fourier Transform.
- We have implemented a new FFT-based relativistic kinetic term.
- The new code accepts arbitrary potential via external files according to a format described on [Comput.Phys.Commun.**272**, 108250]
- We allow for dumping full wave-functions as snapshots. This is specially valuable for extracting excited states.
- Small additional changes: cleanup of the code; we do not require an additional MPI node for controlling,...

Conclusions: extensions to quantumfddd

- We have extended the previous code quantumfddd v.2 to quantumfddd v.3.
- We have implemented 2 additional non-relativistic kinetic terms, that are based on Fast Fourier Transform.
- We have implemented a new FFT-based relativistic kinetic term.
- The new code accepts arbitrary potential via external files according to a format described on [Comput.Phys.Commun.**272**, 108250]
- We allow for dumping full wave-functions as snapshots. This is specially valuable for extracting excited states.
- Small additional changes: cleanup of the code; we do not require an additional MPI node for controlling,...

Conclusions: extensions to quantumfdd

- We have extended the previous code quantumfdd v.2 to quantumfdd v.3.
- We have implemented 2 additional non-relativistic kinetic terms, that are based on Fast Fourier Transform.
- We have implemented a new FFT-based relativistic kinetic term.
- The new code accepts arbitrary potential via external files according to a format described on [Comput.Phys.Commun.**272**, 108250]
- We allow for dumping full wave-functions as snapshots. This is specially valuable for extracting excited states.
- Small additional changes: cleanup of the code; we do not require an additional MPI node for controlling,...

Conclusions: extensions to quantumfdd

- We have extended the previous code quantumfdd v.2 to quantumfdd v.3.
- We have implemented 2 additional non-relativistic kinetic terms, that are based on Fast Fourier Transform.
- We have implemented a new FFT-based relativistic kinetic term.
- The new code accepts arbitrary potential via external files according to a format described on [Comput.Phys.Commun.**272**, 108250]
- We allow for dumping full wave-functions as snapshots. This is specially valuable for extracting excited states.
- Small additional changes: cleanup of the code; we do not require an additional MPI node for controlling,...

Conclusions: numerical stability

- The ground-state wave-function is stable for high values of the time evolution parameter τ .
- The overlap method is an approximation to project out the first and second excited states from the full wave-function.
- For sufficiently high values of τ , the excited states contribution will fall below the machine precision, eliminating the information about excited states.
- The parity projection scripts are a useful tool for separating excited states when they are nearly degenerated in energy which is especially noticeable for the 2s and the 2p-states.

Conclusions: numerical stability

- The ground-state wave-function is stable for high values of the time evolution parameter τ .
- The overlap method is an approximation to project out the first and second excited states from the full wave-function.
- For sufficiently high values of τ , the excited states contribution will fall below the machine precision, eliminating the information about excited states.
- The parity projection scripts are a useful tool for separating excited states when they are nearly degenerated in energy which is especially noticeable for the 2s and the 2p-states.

Conclusions: numerical stability

- The ground-state wave-function is stable for high values of the time evolution parameter τ .
- The overlap method is an approximation to project out the first and second excited states from the full wave-function.
- For sufficiently high values of τ , the excited states contribution will fall below the machine precision, eliminating the information about excited states.
- The parity projection scripts are a useful tool for separating excited states when they are nearly degenerated in energy which is especially noticeable for the 2s and the 2p-states.

Conclusions: numerical stability

- The ground-state wave-function is stable for high values of the time evolution parameter τ .
- The overlap method is an approximation to project out the first and second excited states from the full wave-function.
- For sufficiently high values of τ , the excited states contribution will fall below the machine precision, eliminating the information about excited states.
- The parity projection scripts are a useful tool for separating excited states when they are nearly degenerated in energy which is especially noticeable for the 2s and the 2p-states.

Conclusions: future work

- Implementing a suitable binary format for the wave-functions.
- Implementing an automatized algorithm in order to look for saddle points of $E(\tau)$
- Implementing new projection operators (for instance, d -wave states). This may imply including contributions from higher states in the initial guess.
- Allowing for variations of the coupling strength of each potential.
- Implementing an implicit Crank-Nicolson method. It makes the evolution unconditionally stable regardless of the choice of time step $\Delta\tau$. Issue: the relativistic Schrödinger equation is a non-linear partial differential equation.
- (Anti)-symmetric solution for a two-particle systems in a background potential with an interaction potential between the particles. Of interest for tetraquark phenomenology.

Conclusions: future work

- Implementing a suitable binary format for the wave-functions.
- Implementing an automatized algorithm in order to look for saddle points of $E(\tau)$
- Implementing new projection operators (for instance, d -wave states). This may imply including contributions from higher states in the initial guess.
- Allowing for variations of the coupling strength of each potential.
- Implementing an implicit Crank-Nicolson method. It makes the evolution unconditionally stable regardless of the choice of time step $\Delta\tau$. Issue: the relativistic Schrödinger equation is a non-linear partial differential equation.
- (Anti)-symmetric solution for a two-particle systems in a background potential with an interaction potential between the particles. Of interest for tetraquark phenomenology.

Conclusions: future work

- Implementing a suitable binary format for the wave-functions.
- Implementing an automatized algorithm in order to look for saddle points of $E(\tau)$
- Implementing new projection operators (for instance, d -wave states). This may imply including contributions from higher states in the initial guess.
- Allowing for variations of the coupling strength of each potential.
- Implementing an implicit Crank-Nicolson method. It makes the evolution unconditionally stable regardless of the choice of time step $\Delta\tau$. Issue: the relativistic Schrödinger equation is a non-linear partial differential equation.
- (Anti)-symmetric solution for a two-particle systems in a background potential with an interaction potential between the particles. Of interest for tetraquark phenomenology.

Conclusions: future work

- Implementing a suitable binary format for the wave-functions.
- Implementing an automatized algorithm in order to look for saddle points of $E(\tau)$
- Implementing new projection operators (for instance, d -wave states). This may imply including contributions from higher states in the initial guess.
- Allowing for variations of the coupling strength of each potential.
- Implementing an implicit Crank-Nicolson method. It makes the evolution unconditionally stable regardless of the choice of time step $\Delta\tau$. Issue: the relativistic Schrödinger equation is a non-linear partial differential equation.
- (Anti)-symmetric solution for a two-particle systems in a background potential with an interaction potential between the particles. Of interest for tetraquark phenomenology.

Conclusions: future work

- Implementing a suitable binary format for the wave-functions.
- Implementing an automatized algorithm in order to look for saddle points of $E(\tau)$
- Implementing new projection operators (for instance, d -wave states). This may imply including contributions from higher states in the initial guess.
- Allowing for variations of the coupling strength of each potential.
- Implementing an implicit Crank-Nicolson method. It makes the evolution unconditionally stable regardless of the choice of time step $\Delta\tau$. Issue: the relativistic Schrödinger equation is a non-linear partial differential equation.
- (Anti)-symmetric solution for a two-particle systems in a background potential with an interaction potential between the particles. Of interest for tetraquark phenomenology.

Conclusions: future work

- Implementing a suitable binary format for the wave-functions.
- Implementing an automatized algorithm in order to look for saddle points of $E(\tau)$
- Implementing new projection operators (for instance, d -wave states). This may imply including contributions from higher states in the initial guess.
- Allowing for variations of the coupling strength of each potential.
- Implementing an implicit Crank-Nicolson method. It makes the evolution unconditionally stable regardless of the choice of time step $\Delta\tau$. Issue: the relativistic Schrödinger equation is a non-linear partial differential equation.
- (Anti)-symmetric solution for a two-particle systems in a background potential with an interaction potential between the particles. Of interest for tetraquark phenomenology.