Implementation Notes of `ffr-LFDFT`

Fadjar Fathurrahman

August 22, 2017

Contents

Chapter 1

Lagrange functions in one dimension

1.1 Introduction

There are various ways to derive what will be referred to as Lagrange basis functions (LBFs) or Lagrange functions (LFs) below. In some references, they are also referred to as discrete variable representation (DVR) basis functions, especially in papers by Tuckerman's research group [references needed].

In summary, this is the parameters that are needed to specify a specific LBFs:

- For periodic and cluster/box LBFs we need to specify:
  - number of basis functions N
  - two end points a and b
- For sinc LBFs, we need to specify:
  - number of basis functions N
  - grid spacing h

1.2 Implementation

Currently, there is only special module to handle global variables related to 1D LFs. However, there is a special module to handle 3D LFs, namely the module `m_LF3d` defined in file `m_LF3d.f90`. The relevant global variables for our current discussion are the following.

```
INTEGER :: LF3d_NN(3)
REAL(8) :: LF3d_LL(3)
REAL(8) :: LF3d_AA(3), LF3d_BB(3)
REAL(8) :: LF3d_hh(3)
```

Note that, these arrays are of size 3, for each x, y, and z component. We will restrict ourself to 1D, and will take only the first element, i.e. the x direction. The grid points are stored in array:

```
REAL(8), ALLOCATABLE :: LF3d_grid_x(:)
```

In the actual code, if there is no name-clash (i.e. no two or more variables with the same name), we usually use the aliases for these global variables, e.g.:

```
USE m_LF3d, ONLY : NN => LF3d_NN, hh => LF3d_hh
```

The relevant subroutines to initialize the grid points for each LFs:

- `init_grid_1d_p()`
- `init_grid_1d_c()`
- `init_grid_1d_sinc()`

1.3 Example: initializing grid points for LFs

The following code fragments initializes 1D periodic LFs:

```
USE m_LF3d, ONLY : grid_x => LF3d_grid_x
IMPLICIT NONE
INTEGER :: N, i
REAL(8) :: L
! Initialize the basis functions
ALLOCATE( grid_x(N) )
CALL init_grid_1d_p( N, -0.5d0*L, 0.5d0*L, grid_x )
WRITE(*,'(1x,A,I5)') 'N = ', N
WRITE(*,'(1x,A,F18.10)') 'L = ', L
WRITE(*,'(1x,A,F18.10)') 'Grid spacing = ', grid_x(2) - grid_x(1)
WRITE(*,*) 'Grid points for periodic LF'
DO i = 1,N
  WRITE(*,'(1x,I5,F18.10)') i, grid_x(i)
ENDDO
DEALLOCATE( grid_x )
```

A complete example code can be found in file `tests/init/ex_init_1d.f90`.

1.4 Example: plotting the basis functions

The relevant subroutines used for this purpose are as follows.

- `eval_LF1d_p()`
- `eval_LF1d_c()`
- `eval_LF1d_sinc()`

Plots of periodic, cluster/box, and sinc LFs are shown in Figure 1.1, 1.2, and 1.3 respectively.
An example program which can produce plot data for these figures can be found in `../../tests/plot_1d/ex_p`
The following Python script were used to plot the resulting data:

```
import numpy as np
import matplotlib.pyplot as plt
import os
from matplotlib import rc
rc('font',**{'family':'serif', 'size':16})
rc('text', usetex=True)
types = ['sinc', 'c', 'p']
NBASIS = 5
for t in types:
```

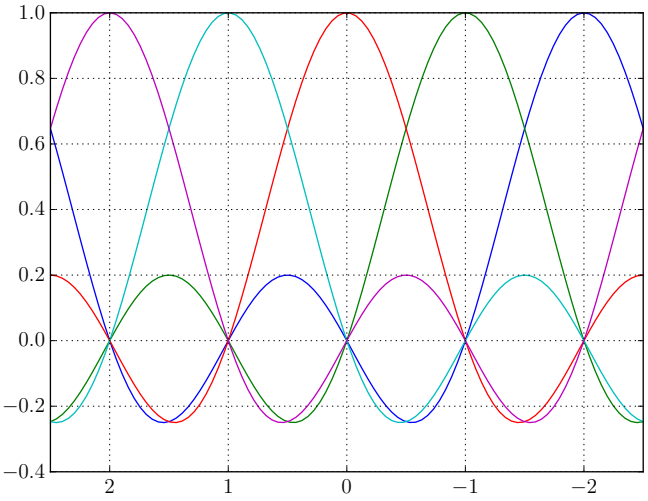

Figure 1.1: Periodic LF

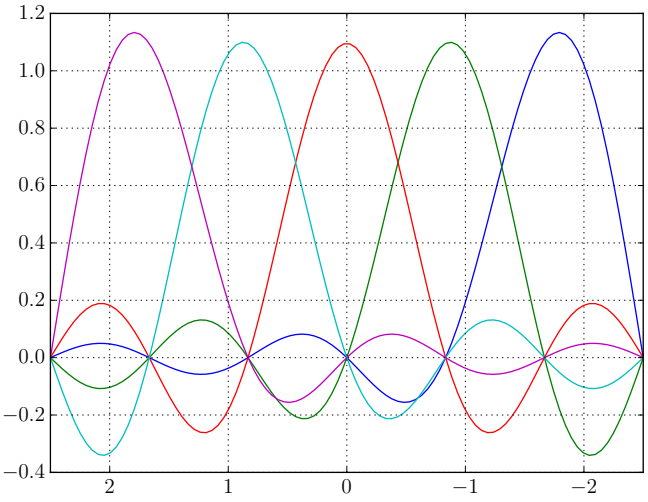

Figure 1.2: Cluster/box LF

```
dat1 = np.loadtxt('LF1d_' + t + '.dat')
plt.clf()
for ibf in range(1,NBASIS+1):
    plt.plot( dat1[:,0], dat1[:,ibf] )
plt.xlim( dat1[-1,0], dat1[0,0] )
plt.grid()
FIPLOT = 'LF1d_' + t + '.pdf'
plt.savefig(FIPLOT)
os.system('pdfcrop ' + FIPLOT + ' ' + FIPLOT)
```

1.5 Example: basis function expansion

1.6 Example: integrals

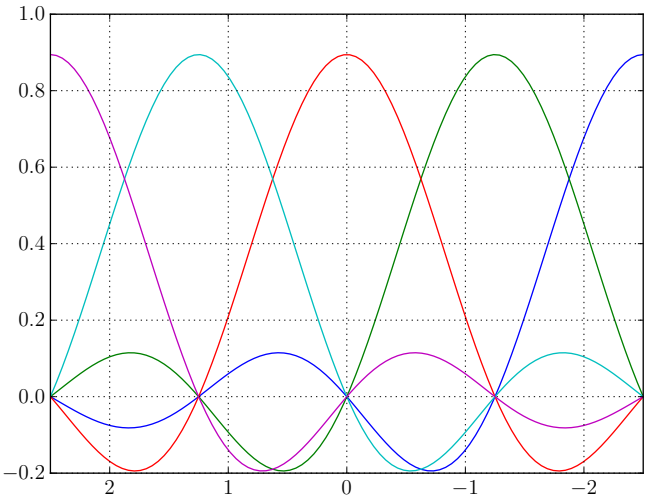

Figure 1.3: Sinc LF

Chapter 2

Solving Schrodinger equation in 1D

Using diagonalization

Chapter 3

Lagrange functions in three dimension

Linear grid

Appendix A

Interpolation with `bspline`

`bspline` library by Jacob Williams.

A.1 Interpolation in 1D

A.2 Interpolation in 3D

Appendix B

Fast Fourier Transform using `fftw3`

3D FFT
C code
Fortran code