ellipFor

Fortran software for the evaluation of Legendre elliptic integrals and Jacobi elliptic functions for generalized input parameters.

Contents

Background

This repository contains files and data for the ellipFor library supporting the article "Algorithm xxx: ellipFor, a Fortran software library for Legendre elliptic integrals and Jacobi elliptic functions with generalized input arguments" by S.J. Trim and R.J. Spiteri.

Variable Definitions

- m = parameter
- ϕ = Jacobi amplitude
- u = first argument of Jacobi elliptic functions
- $K(m)$ = complete Legendre elliptic integral of the first kind
- $E(m)$ = complete Legendre elliptic integral of the second kind
- $F(\phi|m)$ = incomplete Legendre elliptic integral of the first kind
- $E(\phi|m)$ = incomplete Legendre elliptic integral of the second kind
- $\mathrm{sn}(u|m)$ = elliptic sine Jacobi elliptic function
- $\mathrm{cn}(u|m)$ = elliptic cosine Jacobi elliptic function
- $\mathrm{dn}(u|m)$ = delta amplitude Jacobi elliptic function

File Description

Headings that follow indicate directories in the ellipFor repository. CAS (Computer Algebra System) generally refers to SageMath 10.4. In the software development phase, SageMath values were confirmed against values from the Wolfram Language Engine Community Edition (see Legal).

ellipFor/

`LICENSE`

- GPL-3.0 license info

`README.md`

- This documentation file (written in Markdown)

ellipFor/source/

`kind_parameters.f90`

- Contains a module with portable kind parameters imported from the `iso_fortran_env` module

`elliptic.f90`

- Contains module procedures that evaluate Legendre elliptic integrals and Jacobi elliptic functions for generalized input parameter ranges
- Input ranges were generalized by combining transformations with calls to routines from `ellipFor/source/xelbdj2_all_routines.f90` and `ellipFor/source/xgscd_routines.f90`

`xelbdj2_all_routines.f90`

- Contains module procedures for the evaluation of associated elliptic integrals of the first, second, and third kinds
- Assumes standard input parameter ranges
- Adapted from routines by Toshio Fukushima (see Legal)

`xgscd_routines.f90`

- Contains module procedures for the evaluation of the Jacobi elliptic functions sn, cn, and dn
- Assumes standard input parameter ranges
- Adapted from routines by Toshio Fukushima (see Legal)

`Makefile`

- Makefile for building the standalone version of ellipFor using gfortran or ifx
- For use with GNU Make

`ellipFor_test_driver.f90`

- Test driver program for the standalone version of ellipFor
- Evaluates $K(m)$, $E(m)$, $F(\phi|m)$, $E(\phi|m)$, $\mathrm{sn}(u|m)$, $\mathrm{cn}(u|m)$, and $\mathrm{dn}(u|m)$ for specified values of m, ϕ, and u

`ellipFor_test_driver`

- Sample executable for the standalone driver program based on `ellipFor/source/ellipFor_test_driver.f90`
- Results from building ellipFor using GNU Make via `ellipFor/source/Makefile` in the terminal
- gfortran 14.2.0 was used

`ellipFor_test_driver.dat`

- Output file produced by executing `ellipFor/source/ellipFor_test_driver`
- Contains data for $K(m)$, $E(m)$, $F(\phi|m)$, $E(\phi|m)$, $\mathrm{sn}(u|m)$, $\mathrm{cn}(u|m)$, and $\mathrm{dn}(u|m)$

`test_material_driver.f90`

- Test material driver program that verifies the accuracy of ellipFor in detail
- Tests $K(m)$, $E(m)$, $F(\phi|m)$, $E(\phi|m)$, $\mathrm{sn}(u|m)$, $\mathrm{cn}(u|m)$, and $\mathrm{dn}(u|m)$ against a large range of CAS reference values and a number of analytical values

`test_material_driver`

- Sample executable for the test material driver program based on `ellipFor/source/test_material_driver.f90`
- Results from building ellipFor using GNU Make via `ellipFor/source/Makefile` in the terminal
- gfortran 14.2.0 was used

`error_complete.dat`

- Output file produced by executing `ellipFor/source/test_material_driver`
- Contains relative error data compared to CAS for $K(m)$ and $E(m)$
- Used in section 6.1 in the article (see Background)

`error_incomplete.dat`

- Output file produced by executing `ellipFor/source/test_material_driver`
- Contains relative error data compared to CAS for $F(\phi|m)$ and $E(\phi|m)$
- Used in section 6.2 in the article (see Background)

`error_functions.dat`

- Output file produced by executing `ellipFor/source/test_material_driver`
- Contains relative error data compared to CAS for $\mathrm{sn}(u|m)$, $\mathrm{cn}(u|m)$, and $\mathrm{dn}(u|m)$
- Used in section 6.3 in the article (see Background)

`ellipFor/source/expected_data/`

`CAS_complete.dat`

- Contains CAS reference data for complete Legendre elliptic integrals
- Used by `ellipFor/source/test_material_program` to verify accuracy

- Used in section 6.1 of the article (see Background)

`CAS_incomplete.dat`

- Contains CAS reference data for incomplete Legendre elliptic integrals
- Used by `ellipFor/source/test_material_program` to verify accuracy
- Used in section 6.2 of the article (see Background)

`CAS_functions.dat`

- Contains CAS reference data for Jacobi elliptic functions
- Used by `ellipFor/source/test_material_program` to verify accuracy
- Used in section 6.3 of the article (see Background)

`ellipFor_test_driver_OG.dat`

- Expected output from `ellipFor/source/ellipFor_test_driver` corresponding to `ellipFor/source/ellipFor_test_driver.dat`
- Assumes the original values of m, ϕ, and u defined in `ellipFor/source/ellipFor_test_driver.f90` are used
- `ellipFor/source/ellipFor_test_driver` checks output against this reference data automatically (if desired – see Standalone)
- Computed using gfortran 14.2.0

`args_complete_elliptic_integrals.dat`

- Randomized values of m used for testing of ellipFor's $K(m)$ and $E(m)$ values
- Used by `ellipFor/source/test_material_program` to verify accuracy
- Used in section 6.1 of the article (see Background)

`args_incomplete_elliptic_integrals.dat`

- Randomized combinations of ϕ and m used for testing of ellipFor's $F(\phi|m)$ and $E(\phi|m)$ values
- Used by `ellipFor/source/test_material_program` to verify accuracy
- Used in section 6.2 of the article (see Background)

`args_Jacobi_elliptic_functions.dat`

- Randomized combinations of u and m used for testing of ellipFor's $\mathrm{sn}(u|m)$, $\mathrm{cn}(u|m)$, and $\mathrm{dn}(u|m)$ values
- Used by `ellipFor/source/test_material_program` to verify accuracy
- Used in section 6.3 of the article (see Background)

Main ellipFor Subroutines

`complete_elliptic_integrals(m,Fc,Ec)`

- Evaluate $K(m)$ and $E(m)$
- Input: `m` $= m \in \mathbb{R}$ with $m \geq 0$
- Output: `Fc` $= K(m) \in \mathbb{C}$ and `Ec` $= E(m) \in \mathbb{C}$

`incomplete_elliptic_integrals(phi,m,F,E)`

- Evaluate $F(\phi|m)$ and $E(\phi|m)$
- Input: `phi` $= \phi \in \mathbb{R}$, and `m` $= m \in \mathbb{R}$ with $m \geq 0$
- Output: `F` $= F(\phi|m) \in \mathbb{C}$ and `E` $= E(\phi|m) \in \mathbb{C}$

`Jacobi_elliptic_functions(u,m,sn,cn,dn)`

- Evaluate $\mathrm{sn}(u|m)$, $\mathrm{cn}(u|m)$, and $\mathrm{dn}(u|m)$
- Input: `u` $= u \in \mathbb{C}$, and `m` $= m \in \mathbb{R}$ with $m \geq 0$
- Output: `sn` $= \mathrm{sn}(u|m) \in \mathbb{C}$, `cn` $= \mathrm{cn}(u|m) \in \mathbb{C}$, and `dn` $= \mathrm{dn}(u|m) \in \mathbb{C}$

All subroutine arguments are of double precision type (`kind=real64` from the `iso_fortran_env` intrinsic module). Note that the source code for the above routines is in `ellipFor/source/elliptic.f90` and examples for calling the subroutines are in `ellipFor/source/ellipFor_test_driver.f90`.

How to Use

Standalone

Can be used to generate values for $K(m)$, $E(m)$, $F(\phi|m)$, $E(\phi|m)$, $\mathrm{sn}(u|m)$, $\mathrm{cn}(u|m)$, and $\mathrm{dn}(u|m)$ using the driver program `ellipFor/source/ellipFor_test_driver.f90`.

1. Specify the desired m, ϕ, and u in `ellipFor/source/ellipFor_test_driver.f90`
   - Examples are shown in `ellipFor/source/ellipFor_test_driver.f90`
   - Note: this program tests output assuming the original values of m, ϕ, and u are used
     - Warnings will occur if custom values are used
     - To disable these warnings, set `test_output=.false.` near line 10 of `ellipFor/source/ellipFor_test_driver.f90`

2. Build the code using GNU Make with `ellipFor/source/Makefile`
   - Navigate to `ellipFor/source` directory in the terminal
   - Use `make` command in the terminal with the rule for the compiler of choice (gfortran or ifx)
     - Linux/Mac/Windows: `$ make gfortran` or `$ make ifx`
     - Note for Intel oneAPI users: the setvars script must be applied before running `make` (e.g., `$ source /opt/intel/oneapi/setvars.sh`)
     - Intel oneAPI version 2025.0.1 or later is recommended
     - gfortran 14.2.0 or later is recommended
     - GNU Make 4.3 or later is recommended
   - the driver programs `ellipFor/source/ellipFor_test_driver` and `ellipFor/source/test_material_driver` will be produced
     - Note that `ellipFor/source/test_material_driver` can be used to automatically test all ellipFor features in detail
   - if desired, the command `$ make clean` will remove build objects while retaining executables

3. Run `ellipFor/source/ellipFor_test_driver` from the command line
   - Linux/Mac: `$ ./ellipFor_test_driver`
   - Windows Command Prompt: `$ start ellipFor_test_driver`
   - This will produce data for $K(m)$, $E(m)$, $F(\phi|m)$, $E(\phi|m)$, $\mathrm{sn}(u|m)$, $\mathrm{cn}(u|m)$, and $\mathrm{dn}(u|m)$ in the output file `ellipFor/source/ellipFor_test_driver.dat`

With Another Code

Can be used to calculate $K(m)$, $E(m)$, $F(\phi|m)$, $E(\phi|m)$, $\mathrm{sn}(u|m)$, $\mathrm{cn}(u|m)$, and $\mathrm{dn}(u|m)$ from within another code. These instructions presume that the other code is written in Fortran. The following steps are guidelines only. The precise procedure may depend on the particular code used.

Fortran

1. Insert calls to the subroutines for Legendre elliptic integrals and Jacobi elliptic functions within the source of the other code (referred to as `other_code.f90` in the following examples) where necessary
   - Examples of how to call the subroutines are shown in `ellipFor/source/ellipFor_test_driver.f90`

2. Link the f90 files from the `ellipFor/source` folder named `kind_parameters.f90`, `xelbdj2_all_routines.f90`, `xgscd_routines.f90`, and `elliptic.f90` to the source for the other code
   - `ellipFor/source/Makefile` can be used as a template to build `other_code.f90` with the ellipFor libraries and build the executable
     - substitute references to `ellipFor_test_driver` with `other_code` in a copy of the `Makefile` provided
     - customize as desired (e.g., compiler options, etc.)
   - use `$ make gfortran` or `$ make ifx` in the terminal to build the executable `other_code` with the desired compiler
   - Note for Intel oneAPI users: the setvars script must be applied before running `make` (e.g., `$ source /opt/intel/oneapi/setvars.sh`)
   - Intel oneAPI version 2025.0.1 or later is recommended
   - gfortran 14.2.0 or later is recommended
   - GNU Make 4.3 or later is recommended

   - Warning: Duplicate variable/routine names may occur
     - Resolve any related compiler errors
     - Verify that the arguments of subroutine calls correspond to the correct values and data types
3. Run the code executable (`other_code`) as usual

Legal

This repository is subject to the GPLv3 license (`ellipFor/LICENSE`).

The routines contained within `ellipFor/source/xelbdj2_all_routines.f90` and `ellipFor/source/xgscd_routines.f90` are adapted from routines by Toshio Fukushima available under the CC BY-SA 4.0 license. Original versions of these routines can be found at http://dx.doi.org/10.13140/RG.2.2.27011.66085 and https://www.researchgate.net/publication/233903220_xgscdtxt_Fortran_program_package_to_compute_the_Jacobian_elliptic_functions_snum_cnum_dnum.

Some initial software testing in the development phase was performed using the Wolfram Language 14.1.0 Engine Community Edition. The production version of ellipFor does not utilize the Wolfram Language Engine.