RunFortObject.h:

```
/* Generated by Interface Builder */

#import <objc/Object.h>

@interface RunFortObject:Object
{
    id  inputForm;
    id  outputForm;
}

- runFortMethod:sender;

@end
```

Fig. 1

RunFortObject.m:

```
/* Generated by Interface Builder */

#import "RunFortObject.h"
#import <appkit/Form.h>         //inserted by hand, not by IB
#import <stdio.h>                        //ditto

@implementation RunFortObject

- runFortMethod:sender          //body of method inserted by hand
{
  int hellosub_();      //a FORTRAN subroutine, translated into C by f2c
  float indata, outdata;
  FILE *infp, *outfp;

/* read input data from FormCell "Start:" */
  indata = [inputForm floatValueAt:0];
  //printf("indata = %f\n", indata);

/* write indata to file "input.data" for FORTRAN subroutine to read */
  infp = fopen("input.data", "w");
  fprintf(infp, "%f\n", indata);
    /* the \n is needed here to avoid Fortran EOF read errors */
  fclose(infp);

/* call the FORTRAN subroutine */
  hellosub_();

/* read outdata from file "output.data", put there by FORTRAN subroutine */
  outfp =fopen("output.data", "r");
  fscanf(outfp, "%f\n", &outdata);
  fclose(outfp);

/* write output data to FormCell "F77 says:" */
  [outputForm setFloatValue:outdata at:0];

  return self;
}

@end
```

Fig. 2

hellosub.f:

```
      subroutine hellosub

      !print *, 'Howdy, from Fortran ...'

      open(unit=11,status='old',name='input.data')
      read(11,100) x
 100  format(f15.7)
      !print *, 'x = ',x
      y = 7.0*x

      open(unit=12,status='old',name='output.data')
      write(12,100) y

      close(unit=11)
      close(unit=12)

      end
```

Fig 3.

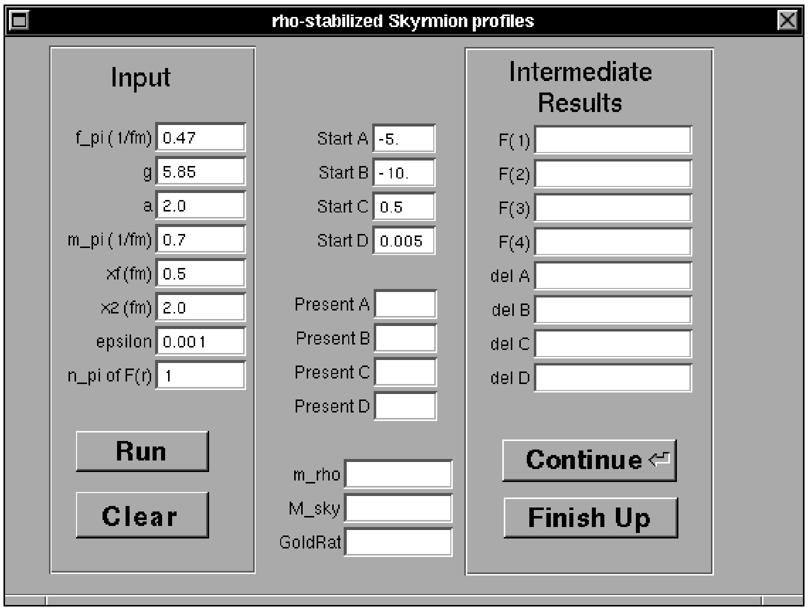
Makefile.preamble:

```
#
# Makefile Preamble for FortFrontEnd.app
#

# declare our other ofiles

OTHER_OFILES = hellosub.o
OTHER_LIBS   = -lf2c
OTHER_DEBUG_LIBS = $(OTHER_LIBS)
```

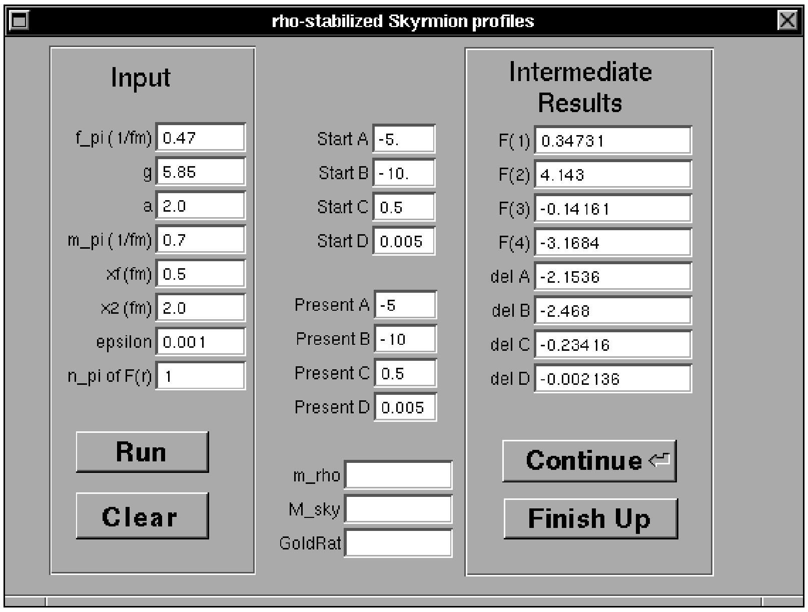
Makefile.postamble:

```
#
# Makefile Postamble for FortFrontEnd.app
#

# other dependencies

hellosub.o: hellosub.f
        f77 -c hellosub.f
```

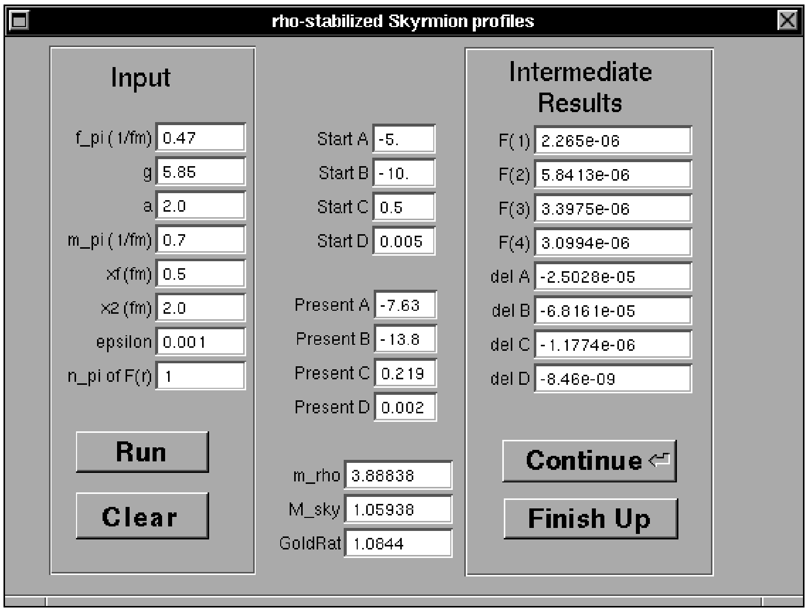
Fig. 4



Main window, as it appears just after the application is launched.

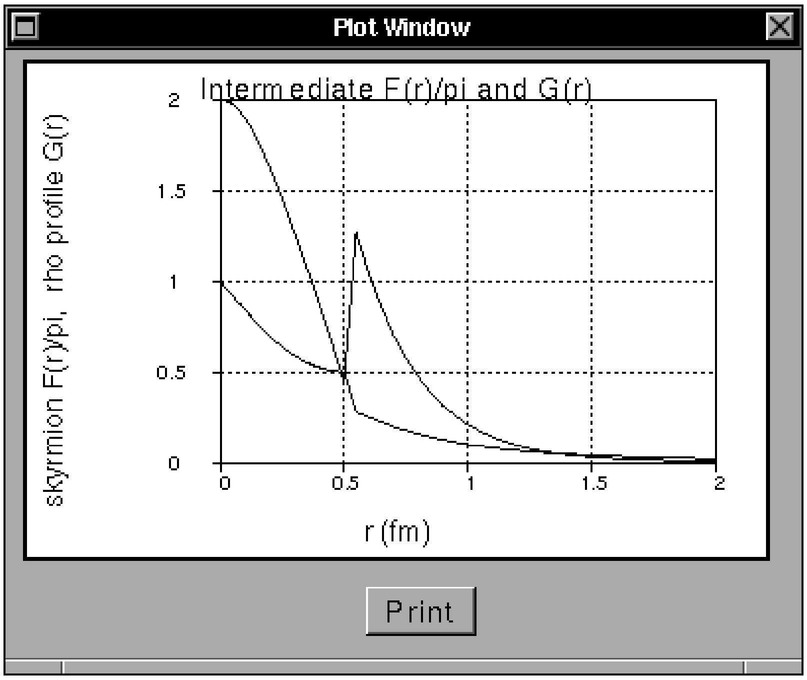


Main window, after the first pass.

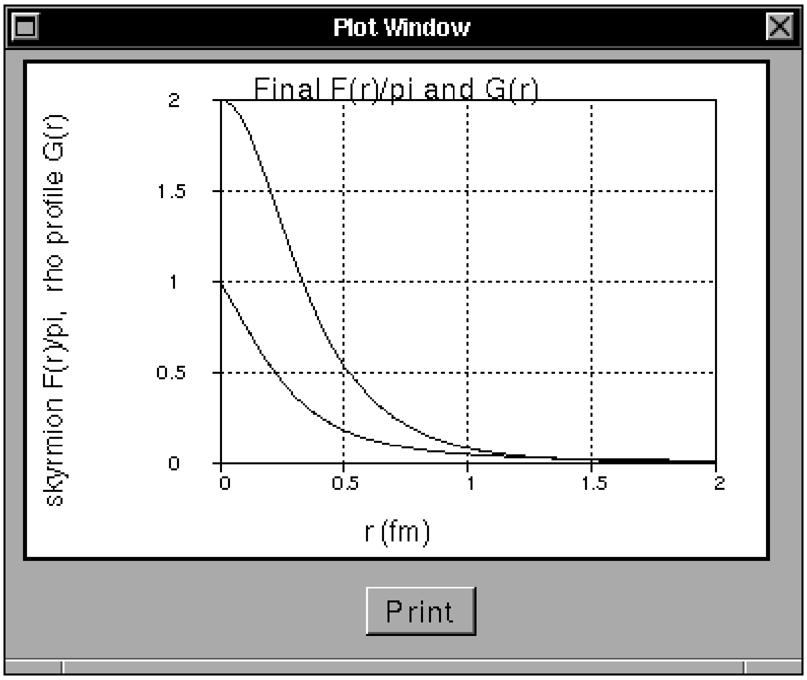


After several iterations and clicking on Finish Up button

Fig. 5



Plot Window, after a typical first pass.



Plot Window, after Finish Up.

Fig. 6

Plotting methods from RunFortObject.m:

```
- updateOutputData:sender
{
  int writeplotdata_();   //a FORTRAN subroutine

  writeplotdata_();
  [self sendPlotDataToWindow];

  return self;
}


- sendPlotDataToWindow
{
  int plotswitch,ii;
  float x[55],F[55],G[55];
  char *PLOT="/Users/silbar/Programming/rhoSkyrmion/plot.data";
  FILE *plotfp;

/*   test output.data file to be sure it can be plotted */
  plotswitch = 1;
  plotfp = fopen(PLOT, "r");
  for (ii=0; ii<55 && plotswitch==1; ii++)
  {
    fscanf(plotfp, "%f%f%f\n", &x[ii],&F[ii],&G[ii]);
    if (abs(F[ii])>10.0 || abs(G[ii])>10.0)
    {
      plotswitch=0;
      //printf("x = %f, F = %e, G = %e\n", x[ii],F[ii],G[ii]);
    }
  }
  fclose(plotfp);
  //printf("plotswitch = %i\n", plotswitch);

  if (plotswitch==1)
    [nxyView plotDataFromStream:self];

  return self;
}


-nxyView:sender provideDataStream:(NXStream **)stream
{
  NXStream *dataStream;
  char *PLOT="/Users/silbar/Programming/rhoSkyrmion/plot.data";

  //printf("\n  Arrive in provideDataStream\n");
  dataStream = NXMapFile(PLOT, NX_READWRITE);
  NXFlush(dataStream);
  *stream = dataStream;

  return self;
}
```

Fig. 7