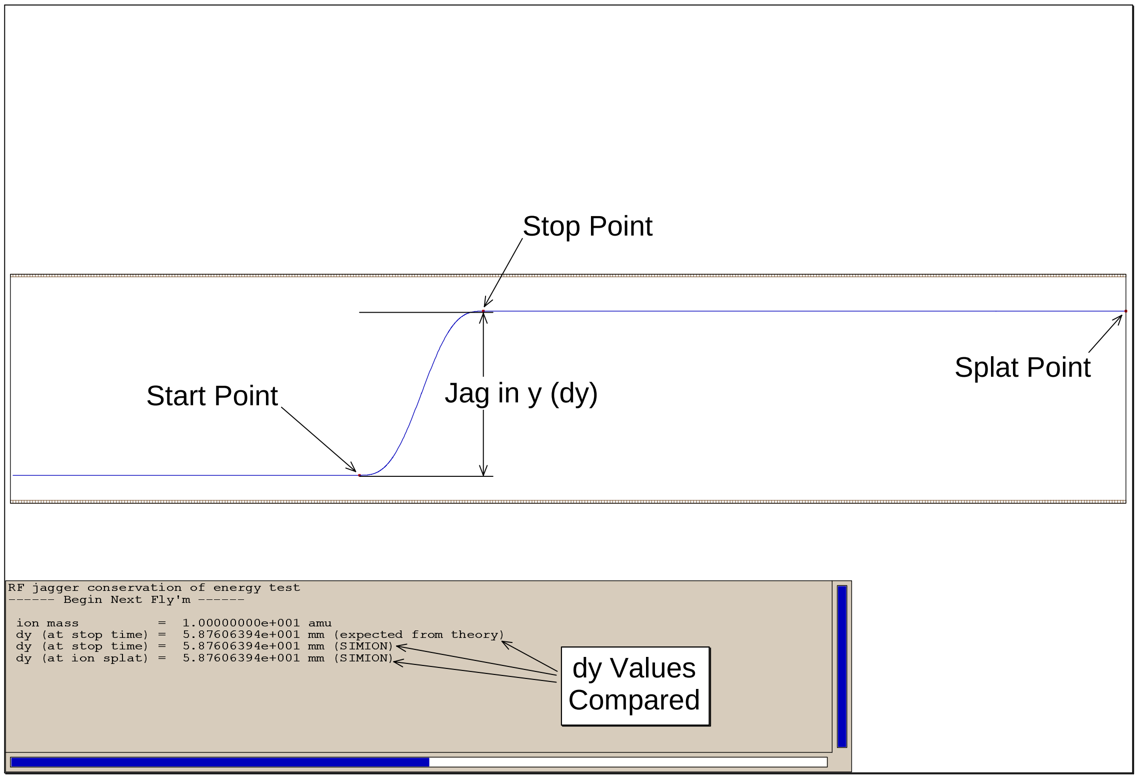
Critical Timing Examples

Introduction

The user program examples in this directory demonstrate how to accurately apply transient potentials to ions. This is a critical problem in certain simulations like bunchers and TOFs. If reality is to be simulated properly then it is crucial that the starting and stopping points of the transients be matched precisely as well has forcing enough time steps in the transient interval to accurately simulate its effects.

In order to get ground truth on the quality of the simulation a rather contrived example is used for the simulations. Ions are sent down co-linear trajectories that are parallel to two large parallel plates. These plates initially have a zero volt potential difference between them. After a delay time, a symmetrical wave (e.g. one cycle of a sine wave) is applied. This wave acts to accelerate the ions in y toward one plate and then decelerate them back to a zero y velocity. If precisely one cycle is used, the ions should jag a predictable distance in y and then resume a trajectory that is precisely parallel to their trajectory before the jag. The quality of the simulation can be judged by how close the jag in y matches the predicted value from first principles and whether or not the velocity in y is really zeroed out so that the y at splat matches the y at the end of the jag (see figure below).



```
RF jagger conservation of energy test
------ Begin Next Fly'm ------

 ion mass            =  1.00000000e+001 amu
 dy (at stop time)   =  5.87606394e+001 mm (expected from theory)
 dy (at stop time)   =  5.87606394e+001 mm (SIMION)
 dy (at ion splat)   =  5.87606394e+001 mm (SIMION)
```

Simulation strategy

The tstep_adjust user program segment is the key to success in these types of simulations. This program segment allows you to monitor and change time steps during the trajectory calculation process. In this case we don't care what the integration time steps are before the start point or after the stop point. However, it is critical that a time step occur precisely on the transient's start point and on its stop point to maintain conservation of energy. Moreover, the spacing of time steps within the interval should be sufficiently dense to obtain an accurate estimate of the forcing envelop. The tstep_user program used in these simulations is shown below:

```
seg tstep_adjust
        rcl rf_frequency 1.0e6 / 1/x ;cycle time in usec
        sto rf_period                                   ;remember rf period
        rcl rf_delay + sto stop_time         ;stop time in usec

        rcl rf_delay
        rcl ion_time_of_flight
        x>=y goto end_check                  ;skip if already beyond rf_delay
        rcl ion_time_step +
        rcl rf_delay
        x>=y exit                                     ;exit if will be less than rf_delay
        rcl ion_time_of_flight -
        sto ion_time_step                    ;force time step to match rf_delay time exactly
        exit

        lbl end_check
        rcl stop_time
        rcl ion_time_of_flight
        x>=y exit                                     ;exit if already beyond stop time

        rcl ion_time_step
        rcl rf_period
        rcl max_rf_fraction_step *
        x<y sto ion_time_step                ;keep time step less than period fraction limit

        rcl ion_time_of_flight
        rcl ion_time_step +
        rcl stop_time
        x>=y exit                                     ;exit if will be less than stop time
        rcl ion_time_of_flight -
        sto ion_time_step                    ;force time step to match stop_time exactly


        exit
```

This segment forces exact start and stop time matching by shortening any time step that would jump over the start or stop point to be just what is needed to hit it precisely. While in the interval, time steps are not allowed to exceed a length dictated by the max_rf_fraction_step (e.g. no larger than 0.001 of the rf period).

Simulation examples

Two simulation examples are provided. The first uses a one cycle sine wave to force the jag. The second reads a waveform in from a data file into an array and uses it to control the transient potentials.

Sine wave simulation

Preparation: Load and refine the array **RF Jagger.pa#**. Remove all PAs. Load the file **RF Jagger.iob** into view and click **Fly'm**.

Seven ions of widely varying masses are flown. The dy displacement (known from first principles) is compared to the dy at the stop points and splats from the simulation. Notice that the agreement is excellent.

Now change the adjustable variable max_RF_fraction_step from 0.001 to 1.001 and re-fly the ions. Notice that the last ion flown has no dy (total error). How can this be? If you examine the trajectory more carefully you may figure it out. First, this is the heaviest ion (slowest). The secret is that it flys less than one grid unit between the transient's start and stop points. This means that SIMION will be pushing the ion along at about one grid unit per time step. Without interval foreshortening time steps will be at the start and stop points and the transient will be completely missed. You can verify this issue by increasing the trajectory quality to 100 (doesn't fix it). However if the trajectory quality is set to 101 (2 steps per grid interval) SIMION will see the transient and model it fairly well.

This serves as an object lesson in how the program works; be suspicious, and protect yourself. If one looked at the first few low mass ions and blithely assumed that all was well with every ion, significant errors could have been made. The use of the above tstep_adjust routine with transients can help avoid some nasty surprises.

An arbitrary waveform simulation

Preparation: Load and refine the array **RF Wave.pa#**. Remove all PAs. Load the file **RF Wave.iob** into view and click **Fly'm**.

The user program loads a user defined waveform data file (**rfwave.dat**). This file contains exactly 45 data points defining relative potentials from the start point (first value) to the stop point (last value). It is assumed (**<u>and</u>** required) that the defined waveform be area symmetric around the 0 potential line.

When the **Fly'm** button is clicked the contents of this file is automatically loaded into the **RF_Wave** array (adjustable variable).

The initialize program segment has two do loops. The first loop scans the array to determine the maximum absolute value for any potential in the array. The second loop uses this value to normalize the array's potentials to be between –1 and +1.

The fast_adjust program segment then divides the transient's time period into 44 intervals. The time location of an ion that is within the transient is determined in terms of these intervals. The ion's interval value is then used along with the normalized array data to determine the normalized potential factor via linear interpolation methods (a spline method could be substituted). This normalized potential is then multiplied times the user specified RF potential to obtain the potential to apply to the plates.

Notice that a smaller max time step size is required to obtain accurate estimates (0.0002 of the transient's period). This is due to the discontinuous nature of the waveform and the sudden slope changes due to the linear interpolation approach.