Pre-requisites:

- Install Python 2.6.x or Python 2.7.x and corresponding versions of NumPy and SciPy
- Add the Python directory to the $PATH environment variable

Test SURVIV on sample input:

Run surviv.py as below to test the SURVIV script.

```
python surviv.py inc.txt surv.txt SURVIV_Result_P.txt
```

Input:

Two input files are required for the SURVIV script.

inc.txt: Each row of this input file contains an alternative splicing event. The 5 columns of this input file contain the read counts for the two isoforms of the alternative splicing event. The read counts for different patients are separated by commas in the column. As an example, for exon skipping events, each row defines a skipped exon and the columns contain the read counts for inclusion and skipping isoforms:

- *ID*: User defined ID for the alternative splicing event.
- *IJC:* inclusion junction counts, patients are separated by comma.
- *SJC:* skipping junction counts, patients are separated by comma.
- *IncFormLen:* length of inclusion form, used for normalization.
- *SkipFormLen:* length of skipping form, used for normalization.

surv.txt: Each row of this input file contains the survival status for a patient. **Important**. **The order of the patients in this file should match the order of patients in inc.txt.** The 3 columns of this input file are:

- *PatientID*: User defined ID for the patient.
- *Time:* Follow up time.
- *Event:* The status indicator, 0=alive, 1=dead.

Output:

For each alternative splicing event, SURVIV outputs the P-values that evaluate the associations between alternative splicing and patient survival.

- *ID*: User defined ID for the alternative splicing event.
- *IJC:* inclusion junction counts, patients are separated by comma.

- *SJC:* skipping junction counts, patients are separated by comma.
- *IncFormLen:* length of inclusion form, used for normalization.
- *SkipFormLen:* length of skipping form, used for normalization.
- *PValue*: P-values of the alternative splicing event.