Envelope opcodes in Csound

Csound has a number of different envelope opcodes. They can be divided into three groups:

(1) linear signal generators
(2) table scanners
(3) envelope shapers

Linear signal generators

These generators trace straight lines or exponential curves between two or more points.

(a) straight lines between two points

```
kr      line    ia, idur, ib
ar      line    ia, idur, ib
```

Trace a line between points **ai** and **bi** in **idur** seconds

(b) straight line segments between more than two points

```
kr      linseg  ia, idur1, ib[, idur2, ic[...]]
ar      linseg  ia, idur1, ib[, idur2, icI...]]
```

Same as before, but any number of segments can be specified.

(c) exponential curves between two points

```
kr      expon   ia, idur, ib
ar      expon   ia, idur, ib
```

Same as (a), but exponential curves, instead of straight lines.

(d) exponential curve segments between more than two points

```
kr      expseg          ia, idur1, ib[, idur2, ic[...]]
ar      expseg          ia, idur1, ib[, idur2, ic[...]]
```

Table scanners

These opcodes can scan tables containing any envelope shapes (normally created using GEN 7 or GEN 5). Table scanners include all oscillators. In special there are two oscillator-type opcodes designed for use as envelope generators:

```
kr      oscil1  idel, kamp, idur, ifn
kr      oscil1i idel, kamp, idur, ifn
```

These two specialised oscillators are designed to scan through a table (number **ifn**) **once** in **idur** seconds. The values read from the table are then multiplied by **kamp**. The start of the process can be delayed by **idel** seconds. The first opcode truncates between values and the second interpolates (more accurate but more expensive in computation). They generate k-rate signals only.
As seen before, we can also use standard oscillators as envelope generators by setting their frequency to 1/duration.

Envelope shapers

Envelope shapers are the standard envelope generators found in Csound, they can both be used as envelope generators or as modifiers.

(a) Linear envelopes

```
kr      linen   kamp, irise, idur, idec
ar      linen   xamp, irise, idur, idec
```

(b) Exponential envelopes

```
kr      envlpx kamp, irise, idur, idec, ifn, iatss, iatdec[,ixmod]
ar      envlpx xamp, irise, idur, idec, ifn, iatss, iatdec[,ixmod]
```

These exponential envelope opcodes are a bit more complex than linen. They are composed of three segments which are:

1) stored function rise shape
2) modified exponential "pseudo steady state"
3) exponential decay

The first segment is read from a function table **ifn** in **irise** seconds. The function table can have any shape, normally starting with a 0 value and finishing with 1 (GEN 5 / 7 can be used again, but other function table generators can also be used). The second stage of the envelope is a pseudo-steady state which will be attenuated/boosted by the value of **iatss**. A value above 1 creates an exponential growth, and a value between 0 and 1 creates an exponential decay, 1 creates a true steady state. The decay stage will last for **idec** seconds and the factor **iatdec** will affect how the output is reduced (generally 0.01). An optional value **ixmod** regulates the exponential growth/decay in the second stage, creating an accelerated (ixmod > 0) or retarded (ixmod < 0) movement towards the target value (values between +0.9 and – 0.9; default 0, no change). Please note also that the function table used for the rise shape has to include what is called an extended guard-point, a number at the end that is the copy of its last value. This means that the size of the table has to be power-of-2 plus 1 (for instance 1025, 2049, 4097, etc).

Exercises:

1) Create an instrument whose amplitude is controlled by a linear envelope and whose frequency vary exponentially from a start value of 200 Hz to a final value of 500 Hz.
2) Create an instrument whose amplitude is controlled by an exponential envelope and whose pitch will jump an octave upwards and back to the original note in the course of its duration. Try two different ways (using different opcodes) of building this instrument.
3) Create an instrument based on the signal flowchart below. Play a few notes with different settings and try to understand what it does. Keep the values for **iratio** (p7) in the range of 0.001 to 0.1.