

## Taller de Música Electrònica

Sergi Jordà

### **Pràctica 1 : Introducció a PD (8 d'abril del 2003-04-07)**

Aquesta pràctica consisteix en realitzar alguns *patches* en PD.

No es imprescindible realitzar tots els *patches* indicats en aquest document, tot i que, evidentment, aquest nombre de *patches* realitzats es tindrà en compte a l'hora d'avaluar aquesta pràctica. Cal afegir també que aquesta pràctica es més llarga de l'habitual. Perquè hi ha les vacances de setmana santa entre mig i perquè es pretén una ràpida introducció a tota la problemàtica específica d'aquest llenguatge gràfic. Per aquestes raons, aquesta pràctica tindrà també un pes superior a l'habitual en l'avaluació final. La pràctica s'haurà d'entregar el 22 d'abril.

Donat que encara no hem vist ni MIDI ni conceptes de síntesi d'àudio, cap dels *patches* indicats a continuació "sona". Tot i així, son *patches* que es podran utilitzar més endavant en la construcció de un possible sistema musical interactiu.

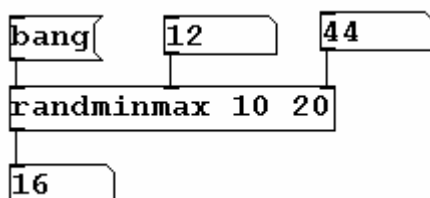
Es recomana per tant completar cada un dels *patches* realitzats amb un petit text explicatiu amb suggeriments sobre les seves possibles aplicacions musicals... amb quins altres *patches* es podrien combinar, quins altres addicionals podria ser interessant tenir, etc. Per pensar en aquestes aplicacions, cal tenir en compte que els números que s'utilitzen podrien ser notes MIDI (valors compresos entre 127) o altres paràmetres MIDI (com ara intensitats, etc., tots ells compresos també normalment entre 0 i 127).

Tot i que ja hi han prou exercicis, si algú veu la possibilitat o l'interès d'ampliar o modificar qualsevol dels *patches* proposats, *feel welcome to do it!!*

Abans o paral·lelament a realitzar els exercicis, es proposa estudiar la documentació inclosa a la web ([Introducció a PD](#) (HTML) o [pdf](#)), la documentació inclosa amb el programa (`doc/1.manual/index.htm`) i estudiar els exemples PD inclosos en el directori `doc/2.control.examples`.

## Propostes:

- Crear un patch **randminmax** que generi nombres aleatoris entre min i max. Min i max seran arguments del patch i també podran modificar-se mitjançant 2 entrades. El patch enviarà un valor aleatori cada cop que rebí un bang per la entrada de la esquerra.



- Crear un patch similar **randcentralrange**, que generi nombres aleatoris al voltant d'un valor central amb un rang de +range.
- **Comptador de bangs**: una entrada (bangs) i una sortida numèrica que s'incrementa cada cop que es rep un bang.
- **Comptador cíclic**: idem que el anterior més una entrada a la dreta (que també estarà com a argument). Quan el comptador arribi a un valor límit tornarà a partir de zero.
- **Conversor de tempo a ms**; aquest patch té 2 entrades i 2 arguments.
  - La entrada de la esquerra serà el valor del tempo (BPM) i la de la dreta, les unitats a dividir (1: negra, 2: corxera, 4: semicorxera...). Qualsevol de les dues entrades, produirà a la sortida un valor en milisegons. Per exemple, en el cas de 60 i 2, el valor de sortida seria 500.
- **Generador de nombres aleatoris predefinits en una taula**.
  - Tindrà 2 entrades (esquerra:bang, dreta: missatge amb llista de valors). Cada cop que es rebí un bang, es sortirà un nombre aleatori inclòs a la taula. Si volem que un nombre tingui més possibilitats de sortir, l'inclourem a la taula varis cops. Els zeros de la taula indiquen "silencis", de forma que si el nombre aleatori resultant fos un zero, el patch no produirà cap sortida (per aquest exercici convé utilitzar arrays).
- **Generador de seqüències semialeatories predefinides en una taula**
  - Es bastant similar a l'anterior. Consta també de 2 entrades (esquerra:bang i dreta: missatge amb llista de valors). Cada cop que es rebí un bang la sortida serà un element correlatiu de la taula. Si la taula contingués per exemple els valors 4 7 8 3 2, la sortida seria 4 7 8 3 2 4 7 8 3 2 4 7....
  - Aquest patch es pot complicar una mica amb una opció addicional (un altre entrada i un altre argument) que admeti un

valor entre 0 i 100. Quan aquest valor sigui 100 (valor defecte) el comportament serà igual que el anterior. Quan sigui 0, el ordre de sortida serà totalment aleatori. Un valor proper a 100, produirà seqüències amb lleugeres variacions sobre l'ordre inicial i les variacions augmentaran conforme disminueixi aquest argument.

- **Filtre i corrector de valors** (o notes musicals).
  - El patch admetrà fins a 11 arguments (compresos entre 0 i 11) i 2 entrades.
    - L'entrada de la esquerra serà el valor (de una nota) a filtrar.
    - L'entrada de la dreta serà una llista per modificar els arguments.
  - L'objecte tindrà 5 sortides (de esquerra a dreta):
    - nota correcta (només es produirà quan la nota d'entrada sigui correcta)
    - nota corregida (i.e. nota inclosa a la taula i la més propera a la de la entrada)
    - octava (de la nota d'entrada)
    - grau o valor de la nota entre 0 i 11
    - nota sense corregir

La sortida 1 es comporta com a filtre, mentre que la sortida 2 es un corrector.

La idea es que aquest patch tregui per exemple un Do (un Re també seria correcta) quan la entrada correspongui a un Do# i la llista d'arguments indiqui la escala de Do Major (0 2 4 5 7 9 11).

Per aquest patch cal tenir en compte que les notes MIDI estan compreses entre 0 i 127 i que els múltiples de 12 sempre corresponen a un Do (0, 12, 24, ...), de forma que si fem %12, les notes amb resta 1 seran un Do#, amb resta 2 un Re, etc...

- **Delay múltiple:** 3 entrades (i 2 arguments) i una sortida
  - Entrada esquerra: valor a retardar
  - Entrada 2 (i argument 1): delay en ms
  - Entrada 3 (i argument 2): nombre de repeticions

Cada cop que es rebí un valor numèric a la entrada, aquest es repetirà per la sortida, retardat i tants cops com s'indiqui.

Si per exemple els arguments fossin 500 i 5, quan es rebés un valor, sortiria immediatament i després 5 cops més retardats entre ells de 500 ms c/u. Exemple:

- Entra un 23
- Sortida 23 a 0 ms
- Sortida 23 a 500 ms
- Sortida 23 a 1000 ms
- Sortida 23 a 1500 ms
- Sortida 23 a 2000 ms

- **Delay amb decaïment:** similar a l'anterior: 3 entrades (i 2 arguments) i una sortida
  - Entrada esquerra: valor a retardar
  - Entrada 2 (i argument 1): delay en ms
  - Entrada 3 (i argument 2): factor decaïment

Cada cop que es rebi un valor, aquest es produirà a la sortida amb el retard indicat i amb valors decreixents.

Si per exemple el arguments fossin 500 i 30, quan es rebés un 100, les sortides serien:

- Sortida 100 a 0 ms
- Sortida 70 a 500 ms
- Sortida 40 a 200 ms

Nota: Mentre el **delay multiple** pot ser útil per entrades de tipus nota, el **delay amb decaïment** pot ser-ho per entrades de tipus intensitat.

- **Quantitzador temporal d'events:** 2 entrades i una sortida
  - Entrada esquerra: bang de trigger
  - Entrada dreta: valor numèric
  - Quan es rebi un bang, la sortida traurà el valor numèric emmagatzemat. Si es tornés a rebre un bang abans de rebre un nou valor numèric, no es produiria cap sortida.
  - Els valors que entren per la dreta no sortiran fins que es rebi un bang per l'esquerra

Es més senzill realitzar el patch sense memòria, de forma que si entre un trigger i un altre es rebessin varis valor numèrics per la dreta, només en sortiria un. En cas contrari caldria definir el tipus de comportament (pila o cua) i el patch es complicaria bastant.

La utilitat musical d'aquest patch pot resultar poc obvia, però es força útil, ja que permet sincronitzar events (ajustant-los a un ritme determinat comú) fent que diferent veus sonin a temps.

- **Quantitzador amb rellotge intern**
  - Una opció alternativa a l'anterior, seria construir aquest sintonitzador amb un rellotge intern. Aquest nou patch, no necessitaria de cap trigger per produir les sortides, tant sols hauria de posar-se en marxa (amb un missatge "start" per exemple) parar-se (amb un missatge "stop") i tindria una segona entrada a la dreta que indicaria el gra temporal (cada quan es pot produir una sortida)
  - Entrada esquerra: valor + "start" + "stop"
  - Entrada dreta: gra o resolució temporal