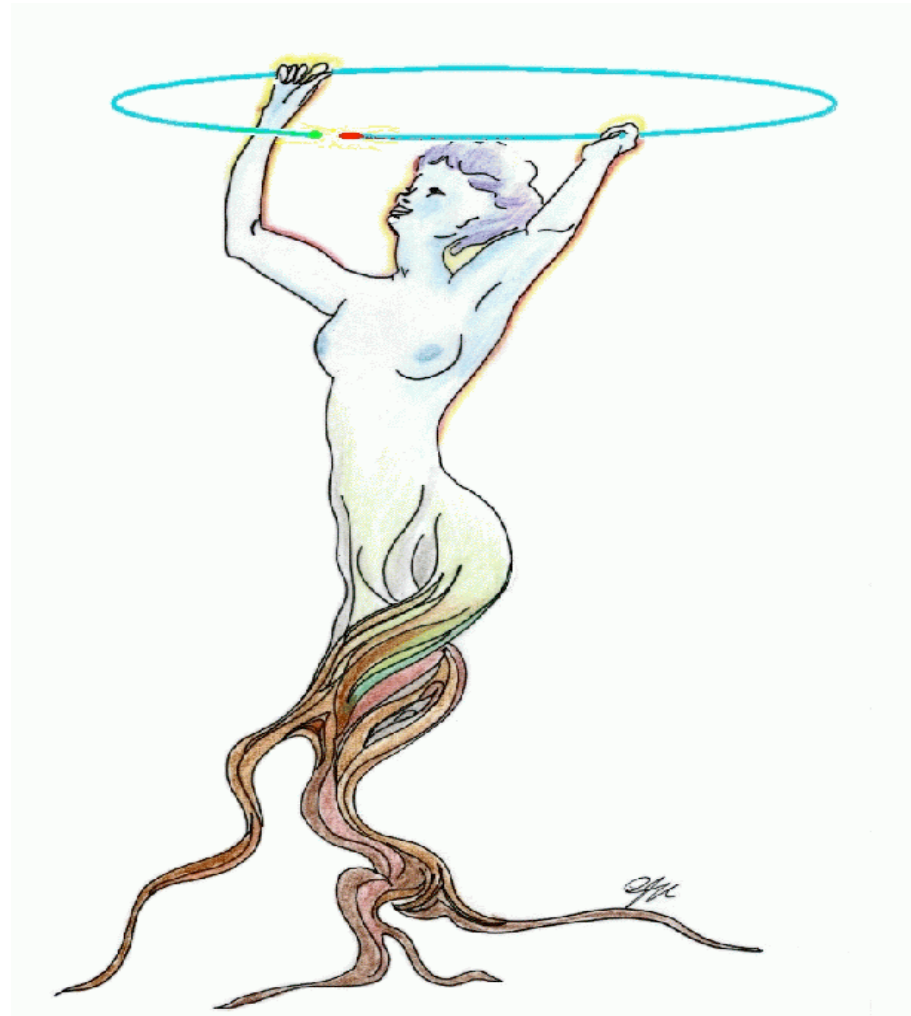


Formation ROOT pour débutants

Quatrième Jour

Les arbres



Grimpons aux arbres...

Aujourd'hui nous allons:

- Le lire
- Faire des analyses
- Sélectionner des évènements
- ...

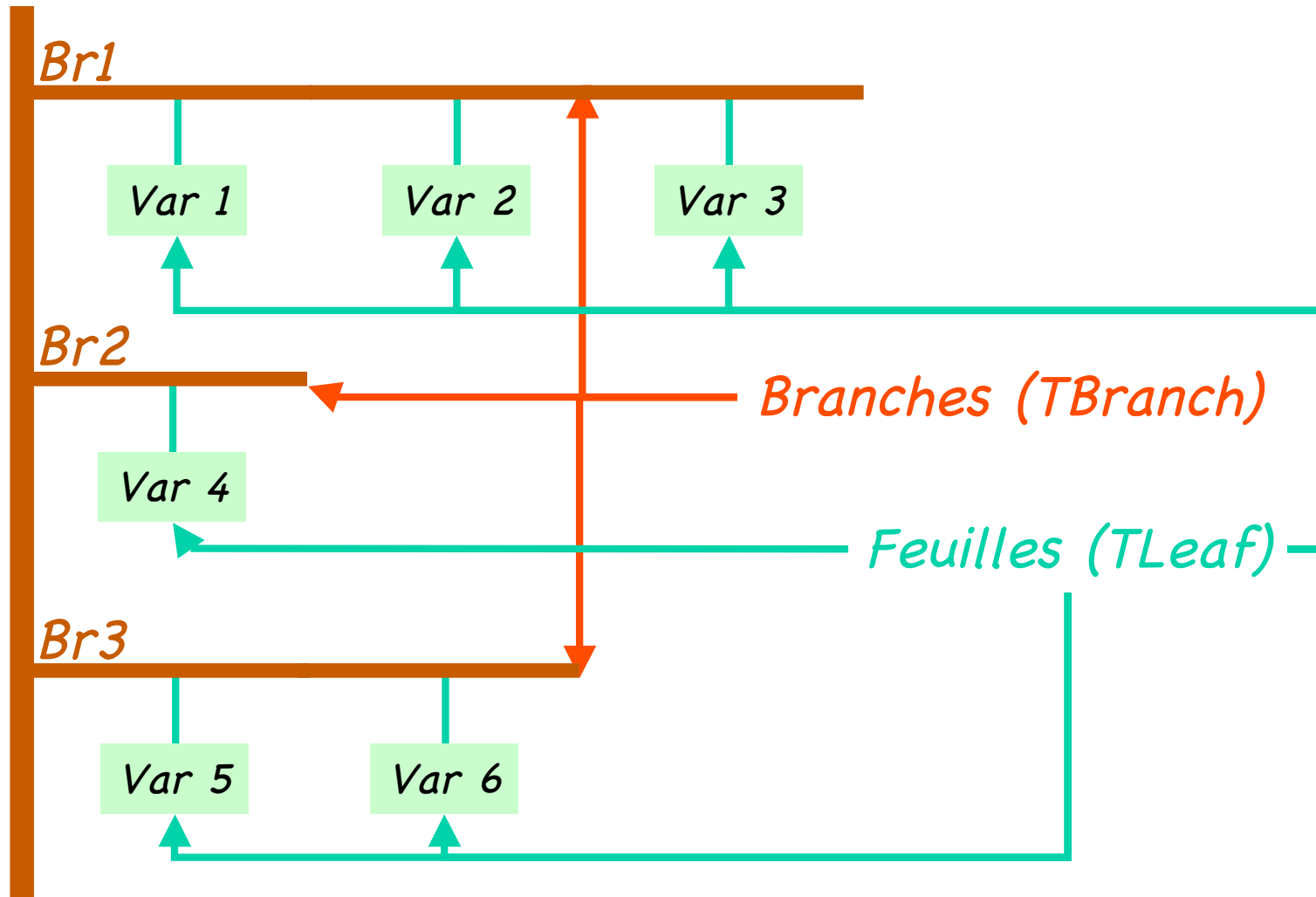


C'est quoi arbre?

- C'est une collection de données organisée de façon rationnelle
 - Chaque événement correspond à une entrée (entry)
 - Chaque entrée a une ou plusieurs branches (TBranch)
 - Chaque branche a une ou plusieurs feuilles (TLeaf, variables)
 - Les feuilles (variables) peuvent être de type:
 - Simple (Float_t, Int_t, Double_t, ...)
 - Structures
 - Classes (on verra demain...)
 - Un tableau de données
- En un mot, c'est un super N-tuple!

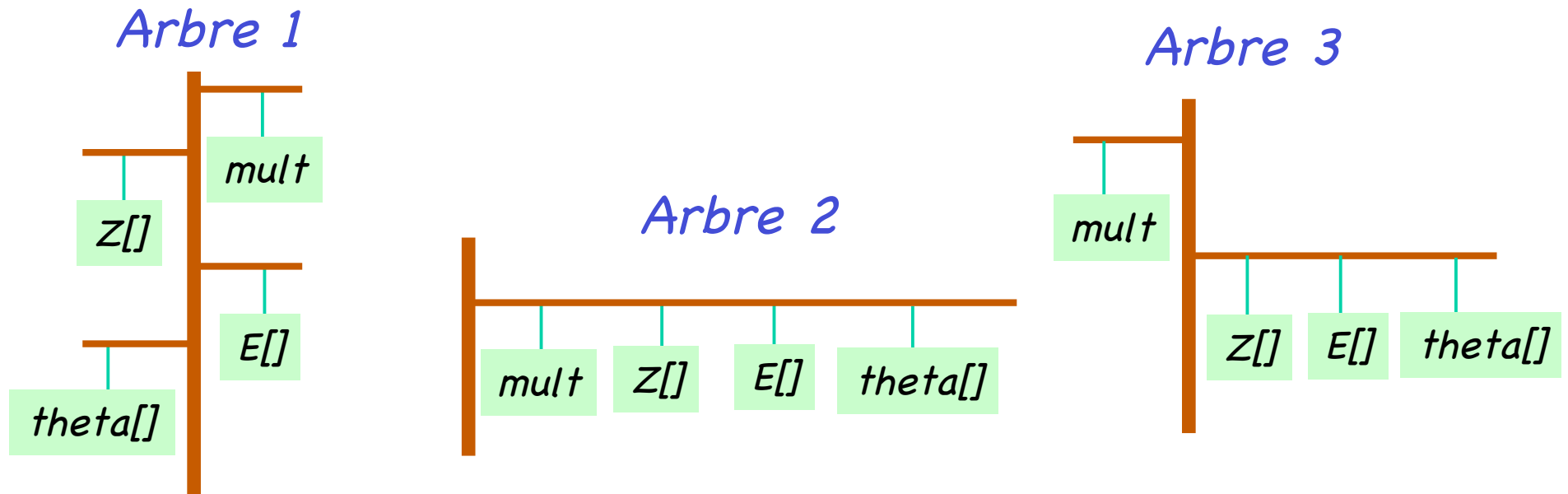
C'est quoi arbre?

Arbre (TTree)



Exemples schématique d'arbres

On veut stocker pour chaque évènement (entry) le nombre de particules (multiplicité) et pour chaque particules la charge Z , l'énergie E et l'angle polaire Θ .



Exemple de fabrication d'arbres dans les fichiers:

http://caeinfo.in2p3.fr/root/Formation/fr/Jour4/tree_struct.C (arbre 1)

http://caeinfo.in2p3.fr/root/Formation/fr/Jour4/tree_struct2.C (arbre 2)

Utiliser un arbre

Ouvrir un fichier avec un arbre

http://caeinfo.in2p3.fr/root/Formation/fr/Jour4/tree_struc.root

- Comment on ouvre un fichier ROOT ?

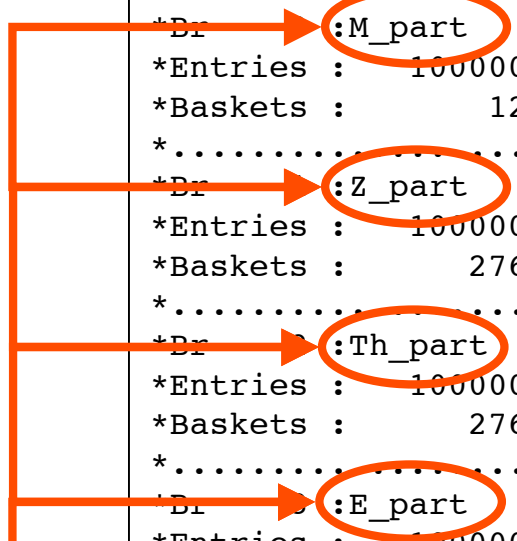
```
root[0] TFile *f=new TFile("tree_struc.root")
root[1] f->ls()
TFile**          tree_struc.root
  TFile*          tree_struc.root
  KEY: TTree      t;1      TTree avec une structure
```

- Comment accéder à l'arbre ?

```
root[2] TTree *a=(TTree *)f->Get("t")
root[3] a->Print()
```

Regarder la structure de l'arbre

```
*****
*Tree      :t          : TTree avec une structure                               *
*Entries   : 100000    : Total =          25750346 bytes File Size = 16900683 *
*          :          : Tree compression factor = 1.52                       *
*****
*Br       :M_part     : Mult/I                                               *
*Entries   : 100000    : Total Size=      401568 bytes File Size =    94299 *
*Baskets   : 12       : Basket Size=     32000 bytes Compression= 4.07      *
*.....*
*Br       :Z_part     : Z[Mult]/F                                           *
*Entries   : 100000    : Total Size=     8449454 bytes File Size =   1840614 *
*Baskets   : 276      : Basket Size=     32000 bytes Compression= 4.58      *
*.....*
*Br       :Th_part    : Theta[Mult]/F                                       *
*Entries   : 100000    : Total Size=     8449745 bytes File Size =    7396565 *
*Baskets   : 276      : Basket Size=     32000 bytes Compression= 1.14      *
*.....*
*Br       :E_part     : Energie[Mult]/F                                       *
*Entries   : 100000    : Total Size=     8449472 bytes File Size =    7520599 *
*Baskets   : 276      : Basket Size=     32000 bytes Compression= 1.12      *
*.....*
```



Nom des branches

Regarder la structure de l'arbre

```
*****
*Tree      :t          : TTree avec une structure *
*Entries  : 100000    : Total =          25750346 bytes File Size = 16900683 *
*          :          : Tree compression factor = 1.52 *
*****
*Br   0 :M_part → Mult/I *
*Entries : 100000    : Total Size=      401568 bytes File Size = 94299 *
*Baskets  : 2        : Basket Size=     32000 bytes Compression= 4.07 *
*.....*
*Br   1 :Z_part → Z[Mult]/F *
*Entries : 100000    : Total Size=     8449454 bytes File Size = 1840614 *
*Baskets  : 276      : Basket Size=     32000 bytes Compression= 4.58 *
*.....*
*Br   2 :Th_part → Theta[Mult]/F *
*Entries : 100000    : Total Size=     8449745 bytes File Size = 7396565 *
*Baskets  : 276      : Basket Size=     32000 bytes Compression= 1.14 *
*.....*
*Br   3 :E_part → Energie[Mult]/F *
*Entries : 100000    : Total Size=     8449472 bytes File Size = 7520599 *
*Baskets  : 276      : Basket Size=     32000 bytes Compression= 1.12 *
*.....*
```

Nom des feuilles

Regarder la structure de l'arbre

```
*****
*Tree      :t          : TTree avec une structure *
*Entries   : 100000    : Total =          25750346 bytes File Size = 16900683 *
*          :           : Tree compression factor = 1.52 *
*****
*Br    0 :M_part      : Mult/I *
*Entries : 100000    : Total Size=      401568 bytes File Size = 94299 *
*Baskets : 12        : Basket Size=     32000 bytes Compression= 4.07 *
*.....*
*Br    1 :Z_part      : Z[Mult]/F *
*Entries : 100000    : Total Size=     8449454 bytes File Size = 1840614 *
*Baskets : 276       : Basket Size=     32000 bytes Compression= 4.58 *
*.....*
*Br    2 :Th_part     : Theta[Mult]/F *
*Entries : 100000    : Total Size=     8449745 bytes File Size = 7396565 *
*Baskets : 276       : Basket Size=     32000 bytes Compression= 1.14 *
*.....*
*Br    3 :E_part      : Energie[Mult]/F *
*Entries : 100000    : Total Size=     8449472 bytes File Size = 7520599 *
*Baskets : 276       : Basket Size=     32000 bytes Compression= 1.12 *
*.....*
```

Type de la variable

<http://root.cern.ch/root/html/TTree.html#TTree:Branch>

Regarder la structure de l'arbre

```
*****
*Tree      :t          : TTree avec une structure *
*Entries   : 100000    : Total =          25750346 bytes File Size = 16900683 *
*          :           : Tree compression factor = 1.52 *
*****
*Br    0 :M_part      : Mult/I *
*Entries : 100000     : Total Size=      401568 bytes File Size = 94299 *
*Baskets : 12         : Basket Size=      32000 bytes Compression= 4.07 *
*.....*
*Br    1 :Z_part      : T[Mult]/ *
*Entries : 100000     : Total Size=      8449454 bytes File Size = 1840614 *
*Baskets : 276        : Basket Size=      32000 bytes Compression= 4.58 *
*.....*
*Br    2 :Th_part     : Theta[Mult]/ *
*Entries : 100000     : Total Size=      8449745 bytes File Size = 7396565 *
*Baskets : 276        : Basket Size=      32000 bytes Compression= 1.14 *
*.....*
*Br    3 :E_part      : Energie[Mult]/ *
*Entries : 100000     : Total Size=      8449472 bytes File Size = 7520599 *
*Baskets : 276        : Basket Size=      32000 bytes Compression= 1.12 *
*.....*
```

Tableau avec la dimension entre []
(taille variable)

Regarder la structure de l'arbre

http://caeinfo.in2p3.fr/root/Formation/fr/Jour4/tree_struc2.root

Nom de la branche

```
*****  
*Tree      :t          : TTree avec une structure *  
*Entries   : 100000    : Total =          60581819 bytes File Size = 18928125 *  
*          :           : Tree compression factor = 3.20 *  
*****  
*Br       0 : bEvent   : Mult/I:Z[50]/F:Theta[50]/F:Energie[50]/F *  
*Entries   : 100000    : Total Size= 60581473 bytes File Size = 18911175 *  
*Baskets   : 1923      : Basket Size= 32000 bytes Compression= 3.20 *  
*.....*  
*****
```

Nom des feuilles

Regarder la structure de l'arbre

http://caeinfo.in2p3.fr/root/Formation/fr/Jour4/tree_struc2.root

Type des variables

```
*****  
*Tree      :t          : TTree avec une structure *  
*Entries   : 100000    : Total = 60581819 bytes File Size = 18928125 *  
*          :           : Tree compression factor = 3.20 *  
*****  
*Br       0 :bEvent    : Mult(I:Z[50]/F:Theta[50]F:Energie[50]F *  
*Entries   : 100000    : Total Size= 60581473 bytes File Size = 18911175 *  
*Baskets   : 1923     : Basket Size= 32000 bytes Compression= 3.20 *  
*.....*  
*****
```

Dimension des tableaux entre []
(taille fixe)

Accès aux données de l'arbre

- Regarder un "événement"

a->Show(15)

```
=====> EVENT:15  
Mult          = 15  
Z             = 30,  
              34, 1, 1, 17, 1,  
              8, 2, 1, 1, 2,  
              2, 1, 1, 2  
Theta        = 14.8766,  
              10.048, 59.2787, 164.868, 8.45649, 21.6054,  
              46.5263, 28.4612, 29.1083, 72.3277, 57.2474,  
              32.4265, 16.6426, 6.97173, 9.6734  
Energie      = 983.813,  
              44.1665, 85.591, 29.5007, 655.211, 59.0234,  
              155.18, 134.403, 21.3786, 10.8284, 19.2134,  
              36.4518, 79.2352, 23.5012, 24.5475
```

Accès aux données de l'arbre

- Trier les événements et afficher certaines variables:

```
a->Scan("Mult:Z[30]:Energie[30]","Mult>30","",1000,0)
```

Sélection 

*Numéro
de l'événement* 

```
*****  
*      Row      *      Mult *      Z[30] * Energie[3 *  
*****  
*          46 *          32 *          2 * 47.778400 *  
*          95 *          31 *          2 * 48.006801 *  
*         399 *          31 *          1 * 28.520700 *  
*         461 *          31 *          2 * 67.939399 *  
*         628 *          32 *          2 * 69.046302 *  
*****  
==> 5 selected entries
```

*L'interface graphique:
le TreeViewer*

Pour l'arbre à une branche

(tree_struc2.root)

The image shows a screenshot of the 'TreeViewer' software interface. The window title is 'TreeViewer' and it has a menu bar with 'File', 'Edit', 'Run', 'Options', and 'Help'. Below the menu bar, there are several controls: a text field labeled 'Option', a 'Histogram' button, a text field with 'htemp', and checkboxes for 'Hist', 'Scan', and 'Rec'. The main area is titled 'Current Tree : t' and displays a tree structure. On the left, there is a 'TreeList' panel with a vertical scrollbar and a 'Coupures' (cuts) section containing several '-empty-' entries. Below this is a 'Bouton de dessin' (drawing button) and a 'STOP' button. On the right, the tree structure is shown with nodes like 'bEvent', 'bEvent.Mult', 'bEvent.Z', 'bEvent.Theta', and 'bEvent.Energie'. Below the tree is a 'Boites à expressions' (expression boxes) section with several '-empty-' entries. At the bottom, there is a 'Mémorisation des commandes' (command storage) section with a progress indicator at '0%' and a 'RESET' button. The status bar at the very bottom shows 'IList', 'OList', and 'First entry : 0 Last entry : 99999'.

Options d'affichage

Amenez les feuilles ici pour tracer des spectres

Coupures

Bouton de dessin

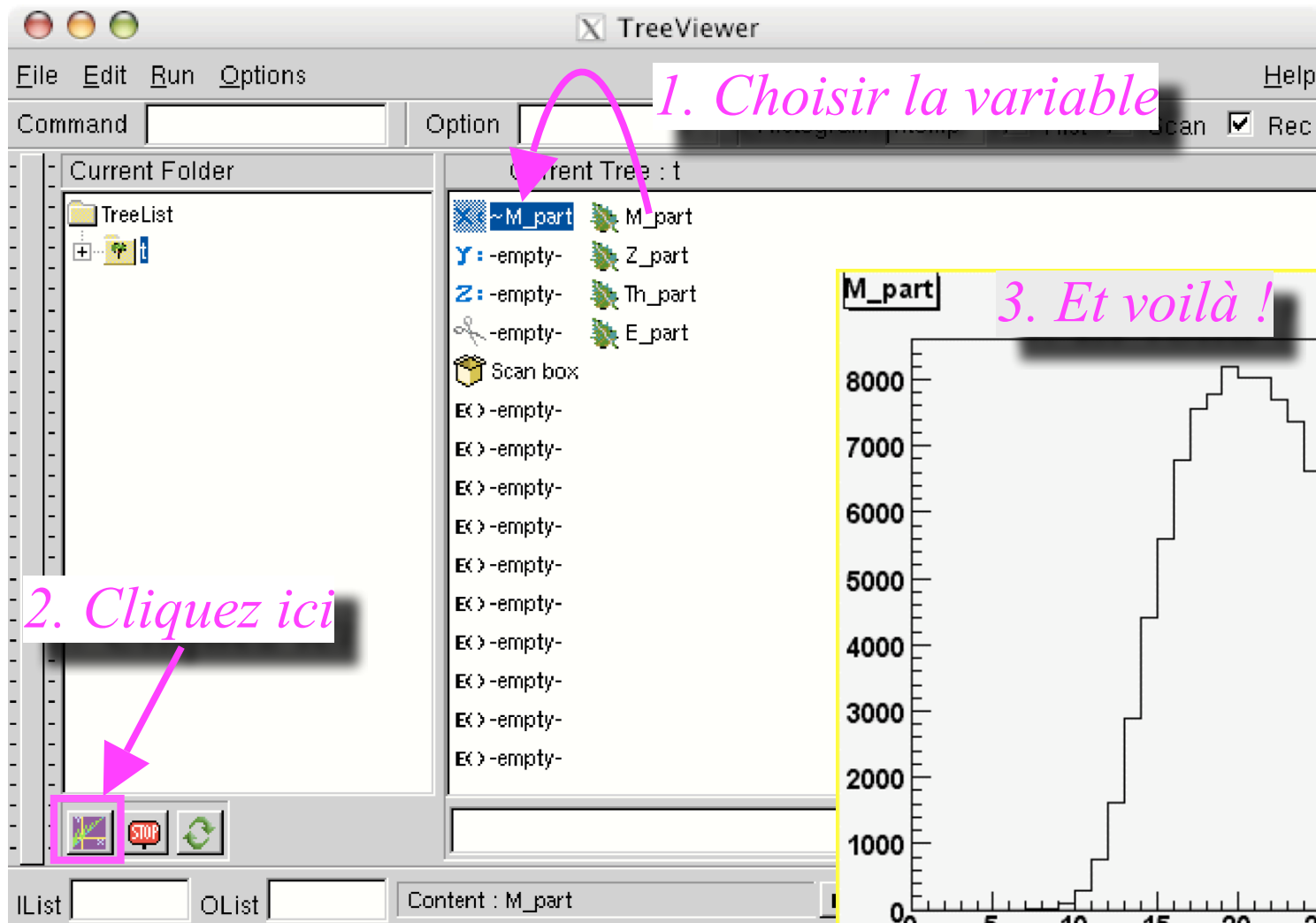
Branche de l'arbre

Les variables (feuilles) de l'arbre

Boites à expressions

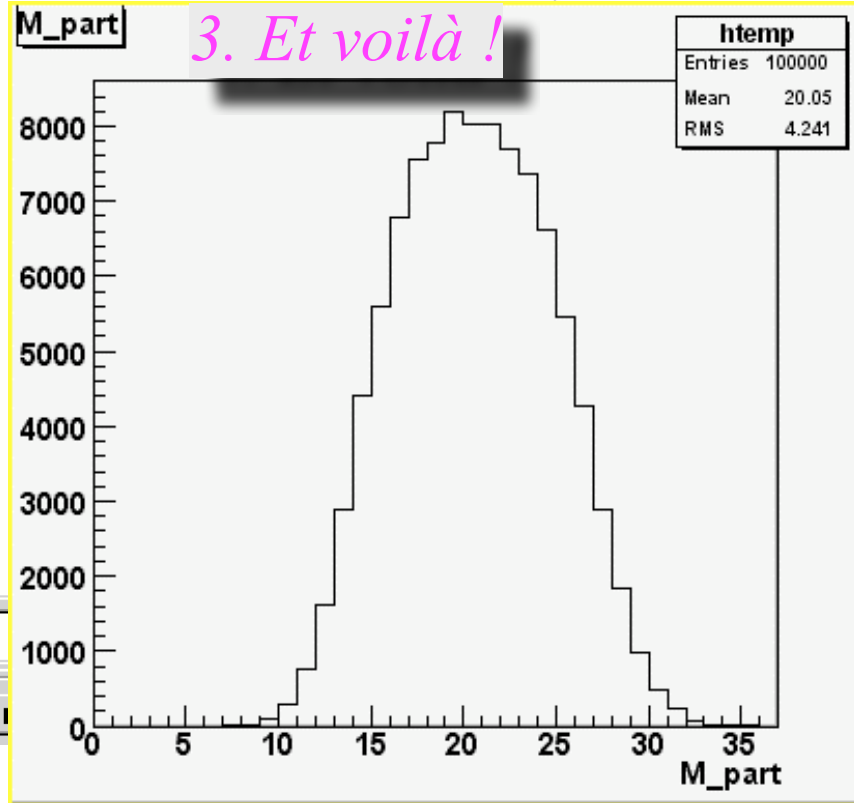
Mémorisation des commandes

Afficher un spectre (1-dim)



1. Choisir la variable

2. Cliquez ici



Afficher un spectre (2-dim)

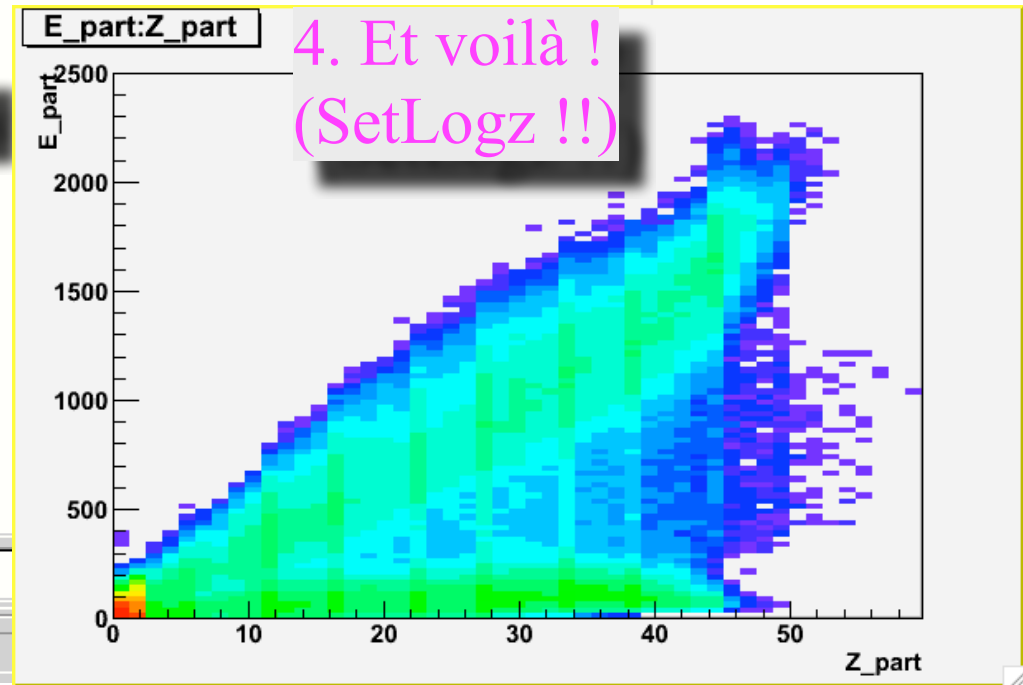
2. Ajuster l'option

The screenshot shows the TreeViewer application window. The 'Option' field is set to 'COL'. The 'Histogram' section has 'htemp' selected, and 'Rec' is checked. The 'Current Tree' list shows several variables: ~Z_part, M_part, ~E_part, Z_part, Z:-empty-, Th_part, -empty-, E_part, and Scan box. The 'Current Folder' pane shows a 'TreeList' folder. The 'Content' field at the bottom is set to 'E_part'. A pink arrow points to the 'Option' field, and another pink arrow points to the 'Z_part' variable in the tree. A third pink arrow points to the 'Plot' button in the bottom left corner.

1. Choisir les variables

3. Cliquez ici

4. Et voilà !
(SetLogz !!)



Mémoriser la commande d'affichage

The image shows a software interface with a tree view on the left and a main workspace on the right. A context menu is open over the tree view, with 'SetRecordName' selected. A dialog box titled 'TTreeView::setRecordName' is open, showing a text field with 'E vs Z' and 'OK'/'Cancel' buttons. A status bar at the bottom shows 'Content : E_part' and a dropdown menu with 'E vs Z' selected. Five numbered annotations in pink text with arrows point to specific elements: 1. 'Cliquez ici' points to a button in the status bar. 2. 'Cliquez droit ici' points to a square in the main workspace. 3. 'Choisir cet item' points to 'SetRecordName' in the context menu. 4. 'Taper le nouveau nom' points to the text field in the dialog box. 5. 'Nom de la mémorisation' points to the dropdown menu in the status bar.

1. Cliquez ici

2. Cliquez droit ici

3. Choisir cet item

4. Taper le nouveau nom

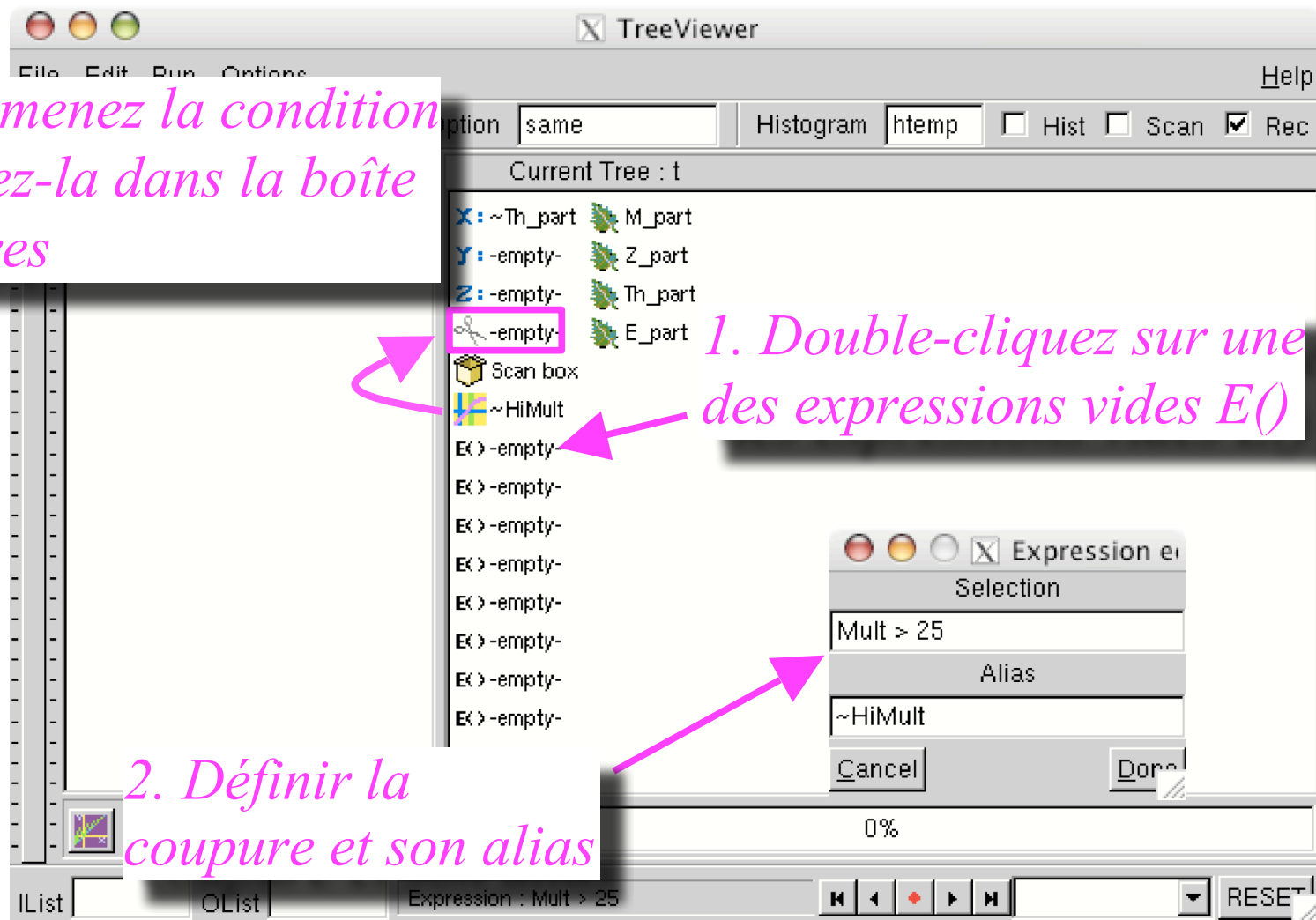
5. Nom de la mémorisation

Utiliser des coupures

3. Puis amenez la condition et déposez-la dans la boîte à coupures

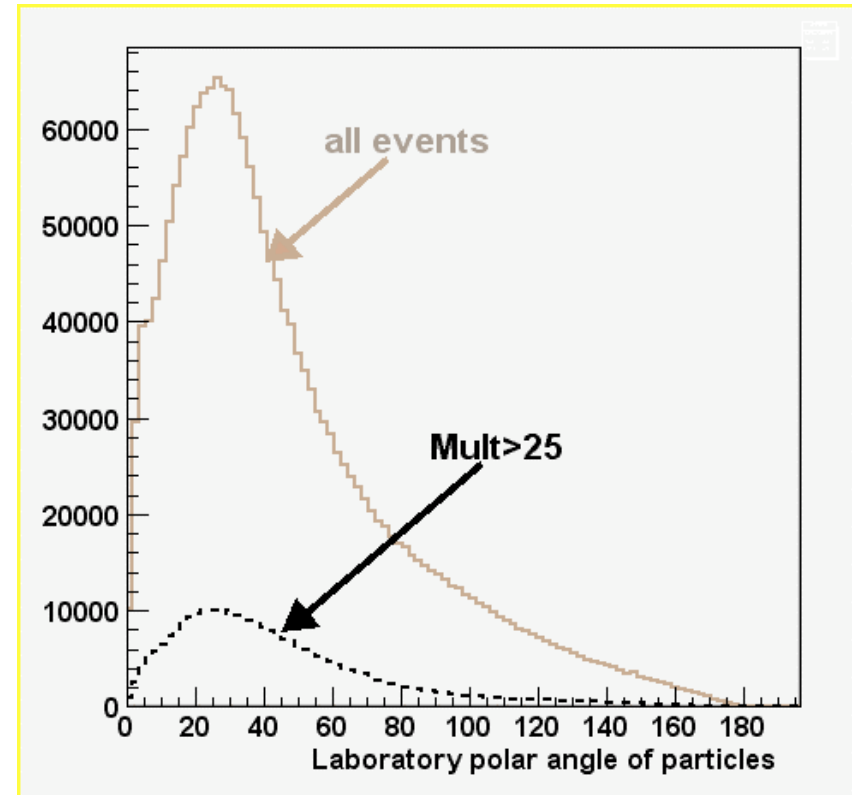
1. Double-cliquez sur une des expressions vides E()

2. Définir la coupure et son alias

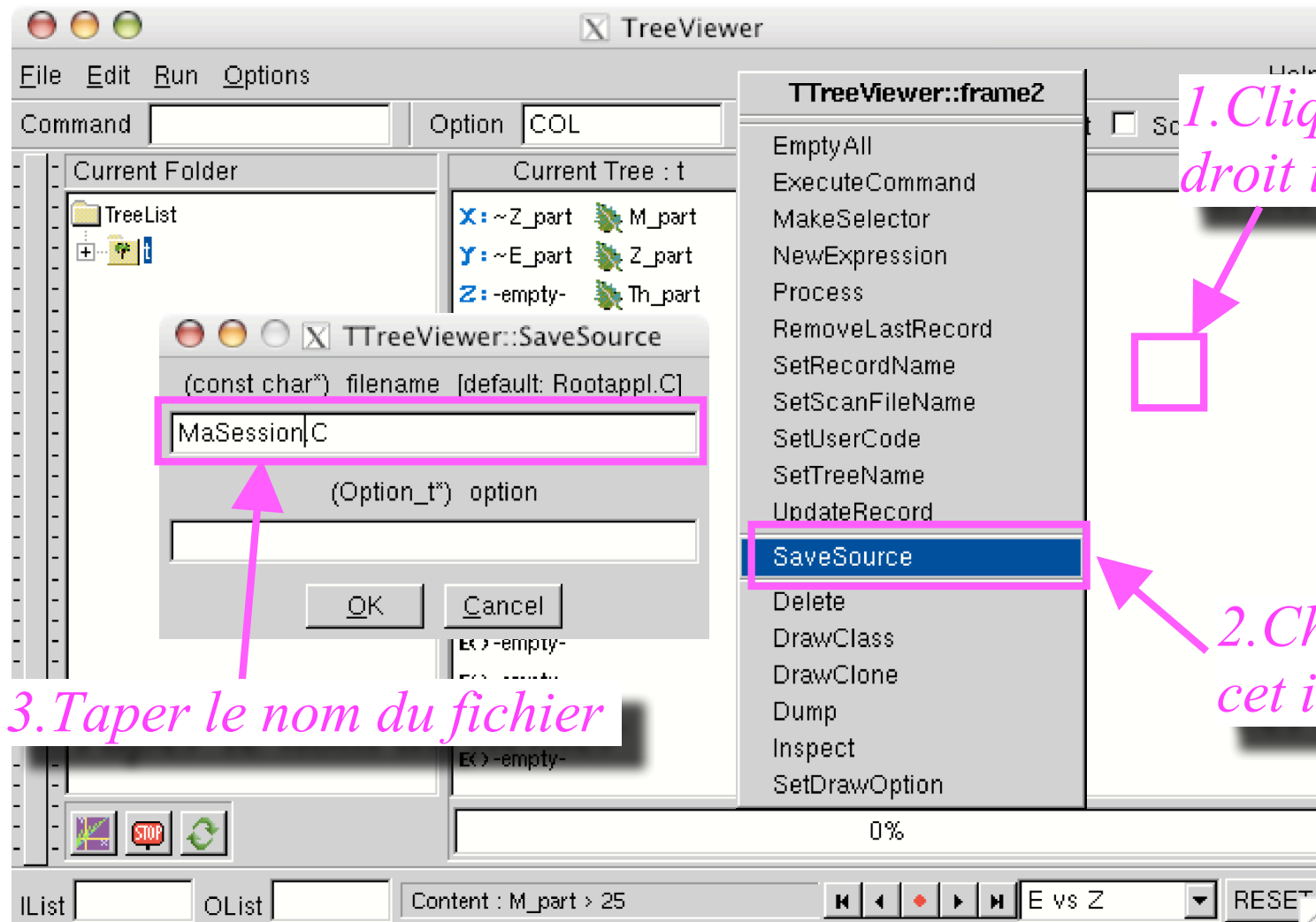


Utilisez des coupures (2)

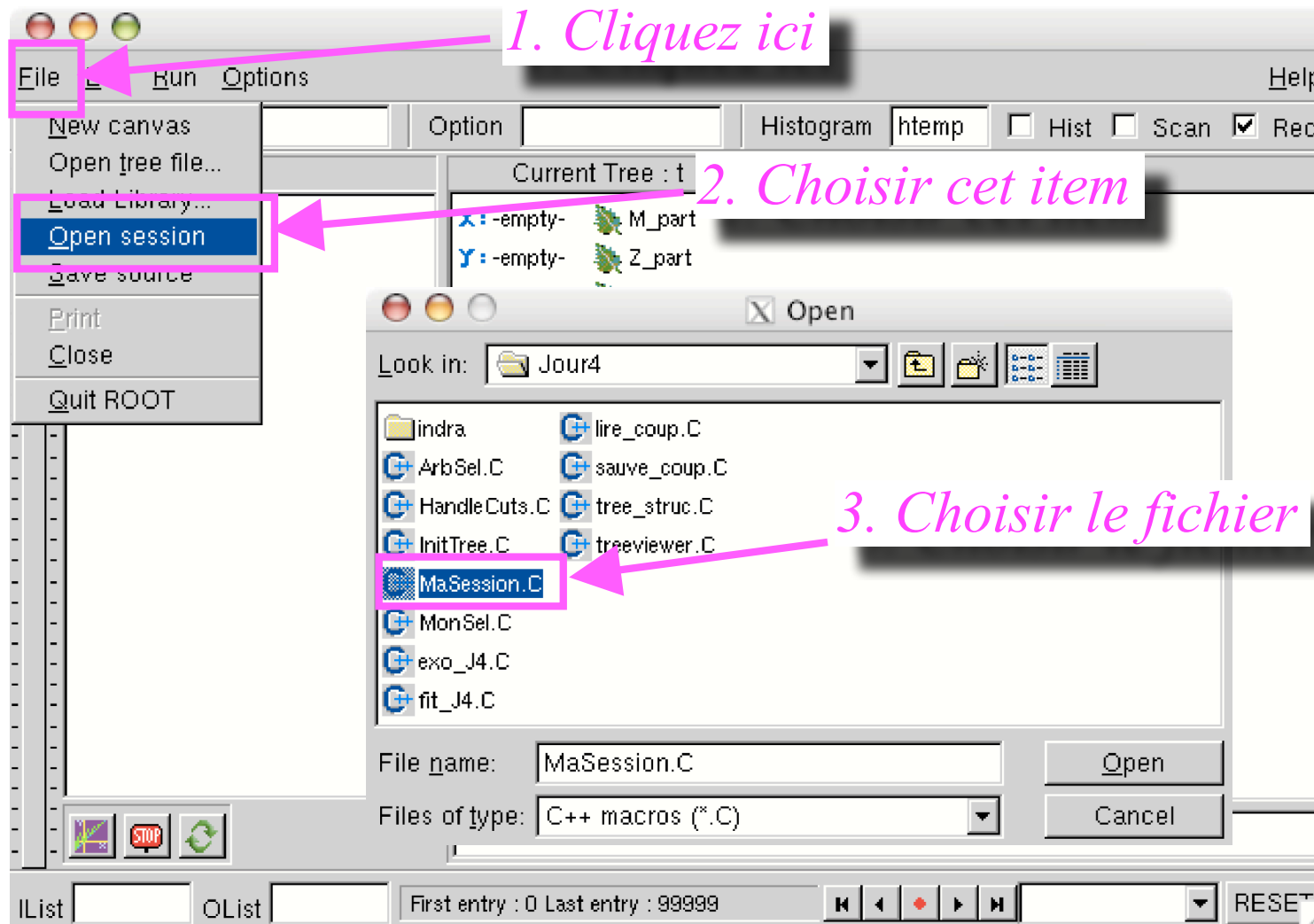
- Amenez la variable **Th_part** sur l'axe x
- Mettez la coupure dans la "boîte à ciseaux"
- Double-cliquez sur la coupure pour la désactiver (barrée rouge)
- Affichez le spectre **sans coupure**
- Ré-activez la coupure
- Tapez "**same**" dans le champ d'options d'affichage
- Affichez le spectre **avec coupure**
- Mémoriser l'affichage
- Soignez la présentation !



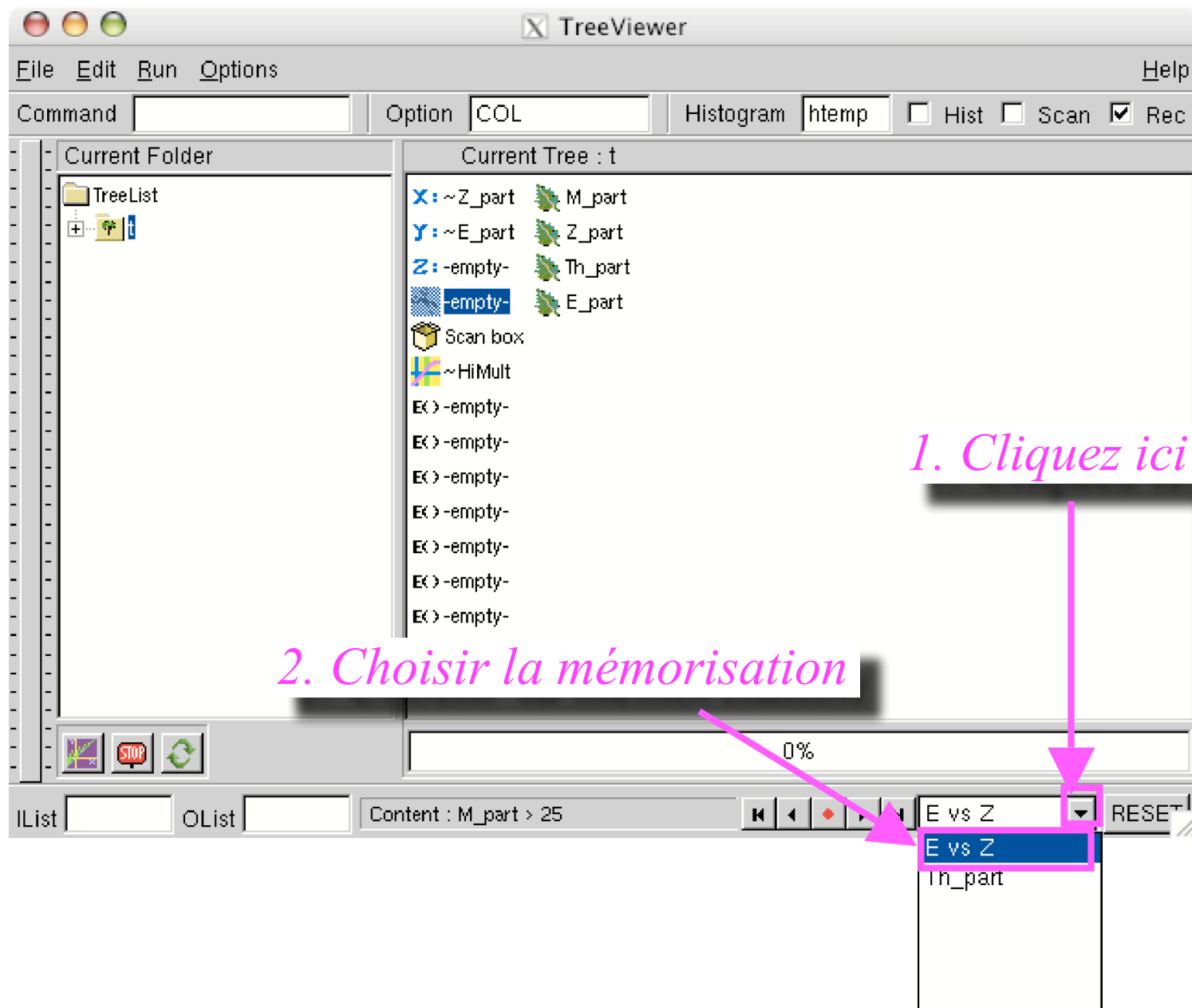
Sauve qui peut...



Tout n'est pas perdu...



Rappel d'une commande mémorisée



*Le hachoir: la machine à
couper*

Coupures graphiques

- Ouvrez la boîte à outils (Canvas menu View->Toolbar)

1. Cliquez ici

2. Dessiner le contour (clique gauche à chaque point, double-clique gauche pour le fermer)

3. Cliquez droit pour activer le menu contextuel

4. Choisir cet item

5. Taper le nom de la coupure

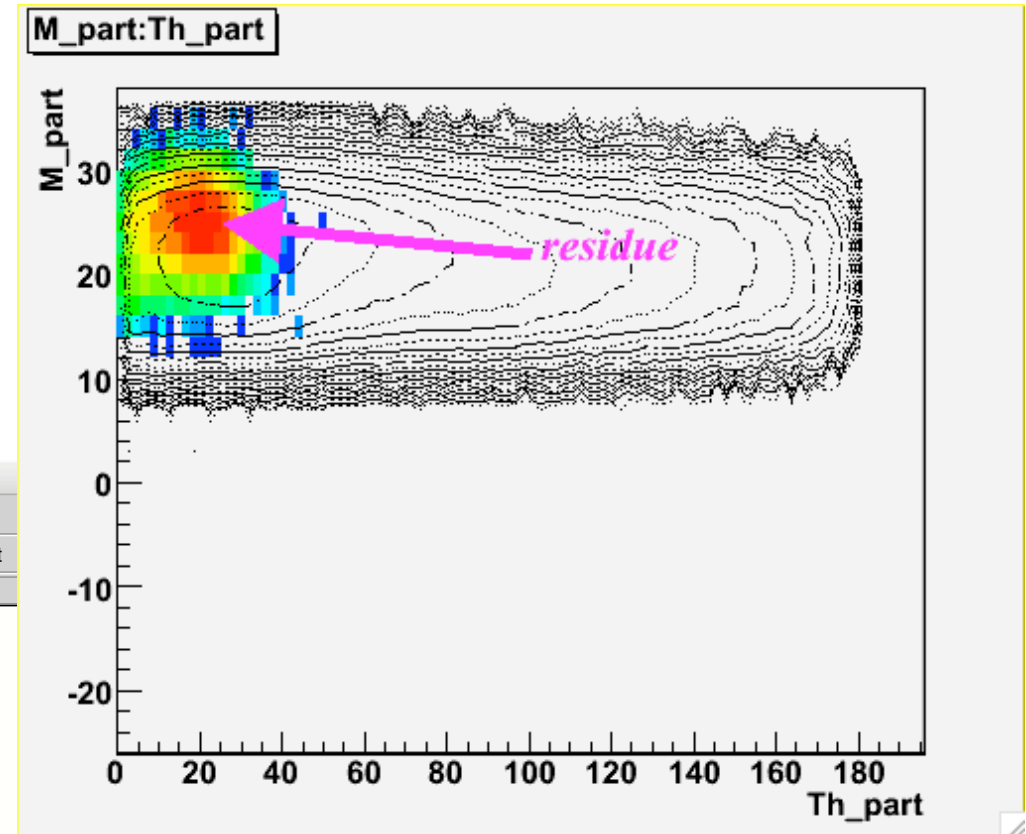
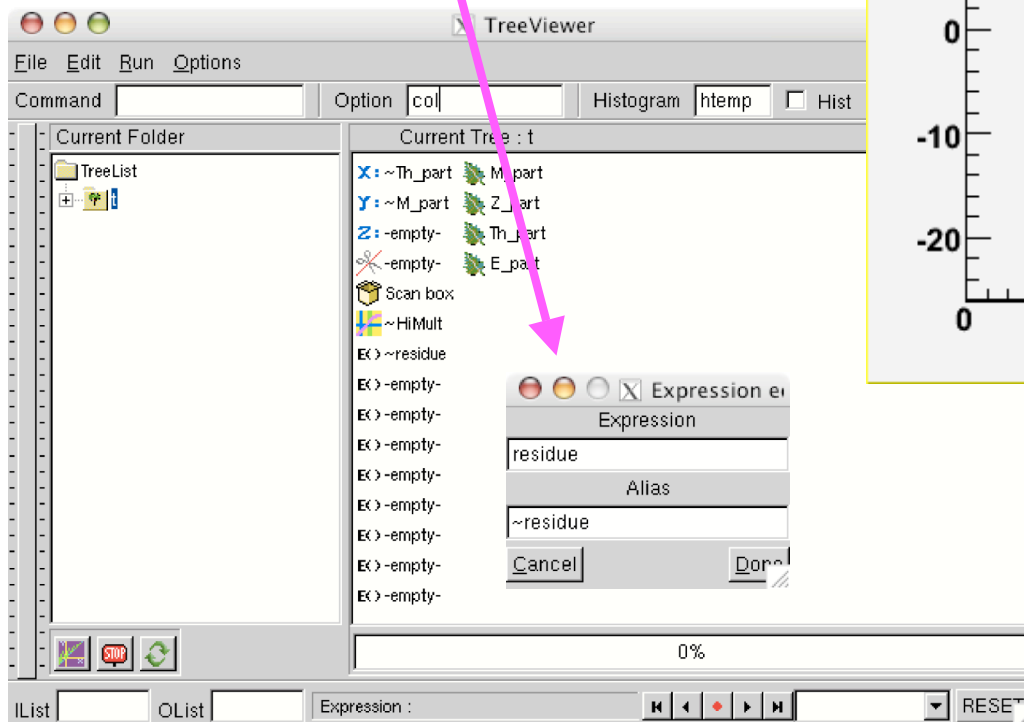
TCutG::SetName
(const char*) name
residue
OK Cancel

art:Z part 1

Z_part

Utilisez la coupure

- Si on rajoute une expression avec la coupure graphique, on peut l'utiliser pour sélectionner les données...



Attention aux doigts: mélangeons les coupures

```
a->Draw("Z_part", "M_part>30", "")
```

- Mais aussi...

```
TCut cut1("M_part > 30")
```

```
a->Draw("Z_part", cut1, "")
```

- Ou encore...

```
TCut cut2("E_part < 200")
```

```
a->Draw("Z_part", cut1 && cut2, "")
```

ET en C++

- Pour les coupures graphiques

```
a->Draw("Z_part", cut1 || "residue", "")
```

OU en C++

Le couteau suisse...

Combinaison de variables

- On peut combiner les variables pour en obtenir d'autres
- Exemples:

afficher la vitesse parallèle V_z

```
a->Draw("sqrt(E_part/(931.5*Z_part))*cos(Th_part*3.1416/180.)")
```

afficher l'énergie transverse en fonction de Z

```
a->Draw("E_part*pow(sin(Th_part*3.1416/180.),2):Z_part", "", "box")
```

- On peut également les définir dans le TreeViewer

Expression	
E_part*pow(sin(Th_part*3.1416/180),2)	
Alias	
~Etrans	
<input type="button" value="Cancel"/>	<input type="button" value="Done"/>

Alias, alias, trois fois alias...

- On peut définir des pseudo variables (alias)

Exemples:

module de la vitesse:

```
a->SetAlias("V","sqrt(E_part/(931.5*Z_part))*30")
```

cosinus de l'angle θ :

```
a->SetAlias("cost","cos(Th_part*3.1416/180.)")
```

composante V_z de la vitesse

```
a->SetAlias("Vz","V*cost")
```

Utilisation:

```
a->Draw("Z_part:Vz","Vz>-10","col")
```

- On peut également les utiliser dans le TreeViewer

ATTENTION: les alias du TreeViewer ne peuvent pas être utilisés avec la commande **a->Draw()**



Bête de somme...

- On peut utiliser des macro-commandes:

Exemples:

Somme des produits $Z*Vz$:

```
a->Draw("Sum$(Z*Vz)")
```

Alias Mimf

```
a->SetAlias("Mimf","Sum$(Z>2)")
```

Z	6	1	4	2	Sum\$(Z>2)
Z>2	1	0	1	0	2

Alias Energie transverse des particules légères

```
a->SetAlias("Et12","Sum$(E*(1-cost*cost)*(Z<=2))")
```

Utilisation:

```
! a->Draw("Mimf:Et12","Sum$(Z>2)>3","col")  
a->Draw("Mimf:Et12","Mimf>3","col")
```

- On peut également les utiliser dans le TreeViewer
- Voir les autres macro-commandes à



Le travail à la chaîne

- On peut utiliser des chaînes de caractère comme arguments de Draw, Scan, SetAlias, GetAlias.

Exemples:

On veut fabriquer les alias "NewVarX" dont la définition est "variableX-(X^{ème} valeur d'un tableau)" pour X allant de 1 à 10

```
Char_t nomAlias[80];
Double_t y[10]={50.3,28.34,10.1,2.02,32.8,33.4,50,33.3,7.24,12.1};
for(Int_t i=1;i<=10;i++)
{
    sprintf(nomAlias,"NewVar%d",i);
    TString var("variable");
    var+=i;
    a->SetAlias(nomAlias,Form("%s-%f",var.Data(),y[i-1]));
}
a->GetListOfAliases()->ls();
```

<http://www.cplusplus.com/ref/cstdio/printf.html>

<http://root.cern.ch/root/html/TString.html>

<http://root.cern.ch/root/html/examples/tstring.C.html>

Projeter dans un histogramme

Création de l'histogramme:

```
TH1F *h1=new TH1F("DistZ",  
                  "Distribution de charge",100,-0.5,99.5)
```

- Projection!

c'est le nom de l'histogramme qui compte!

```
a->Draw("Z_part >> DistZ", "M_part>30")
```

ou

```
a->Project("DistZ", "Z_part", "M_part>30")
```

- Cumuls !

```
a->Draw("Z_part >>+DistZ", "M_part<=30")
```

ou un "+" devant le nom de l'histogramme dans le TreeViewer



Les listes d'évènements

- Ces listes permettent de gagner du temps si on applique une coupure régulièrement: on enregistre les numéros des évènements correspondant à la coupure!

```
a->Draw(">> listem", "M_part>30", "")
```

- Puis

```
TEventList *lm=(TEventList *)gROOT->FindObject("listem")
```

```
lm->Print("all") ← Liste des numéros d'évènements dans la liste
```

```
a->SetEventList(lm)
```

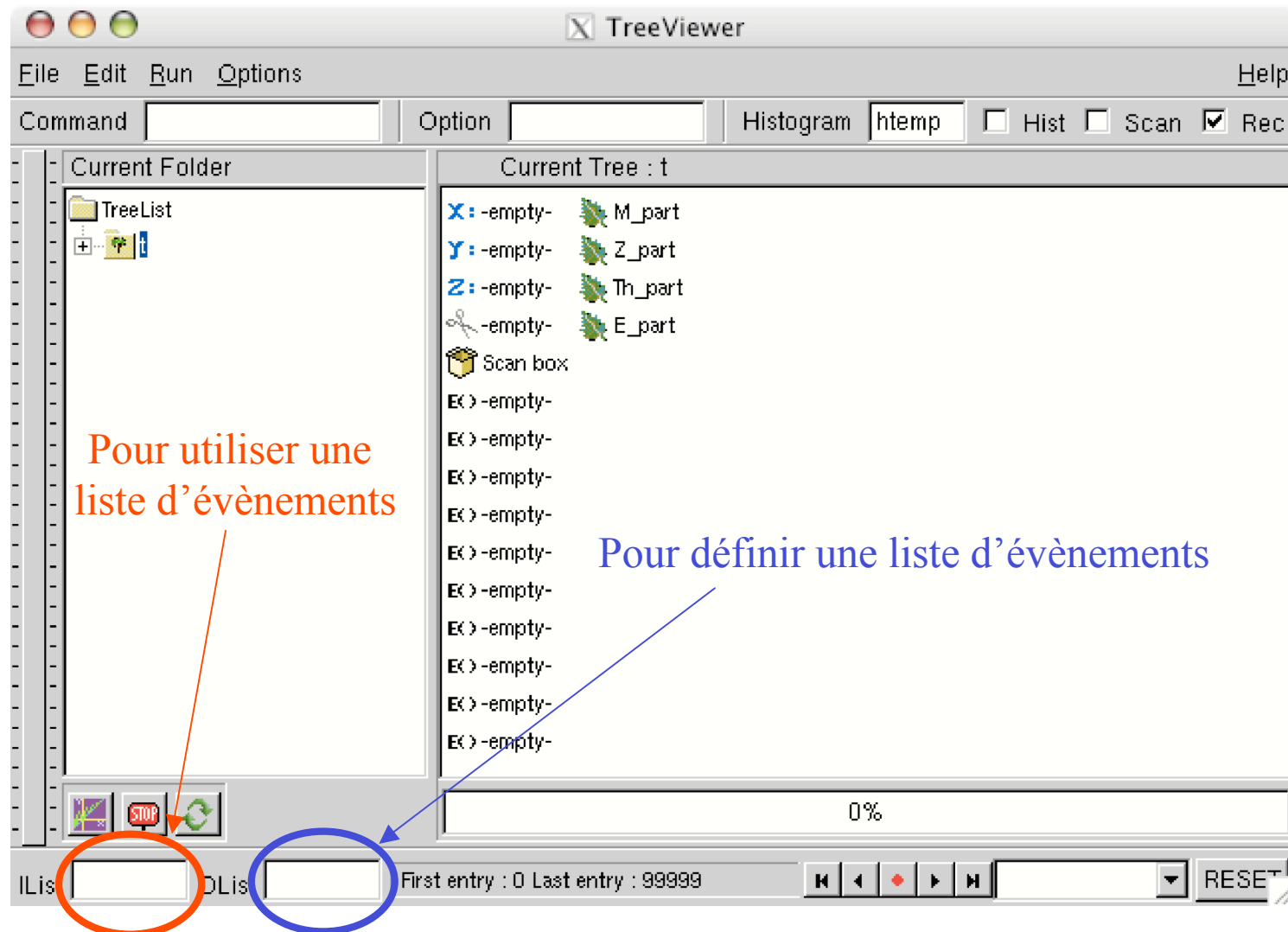
```
a->Draw("Z_part")
```

```
a->Draw("E_part")
```

- Pour ne plus la prendre en compte:

```
a->SetEventList(0)
```

Les listes d'événements avec le TreeViewer



Exercice

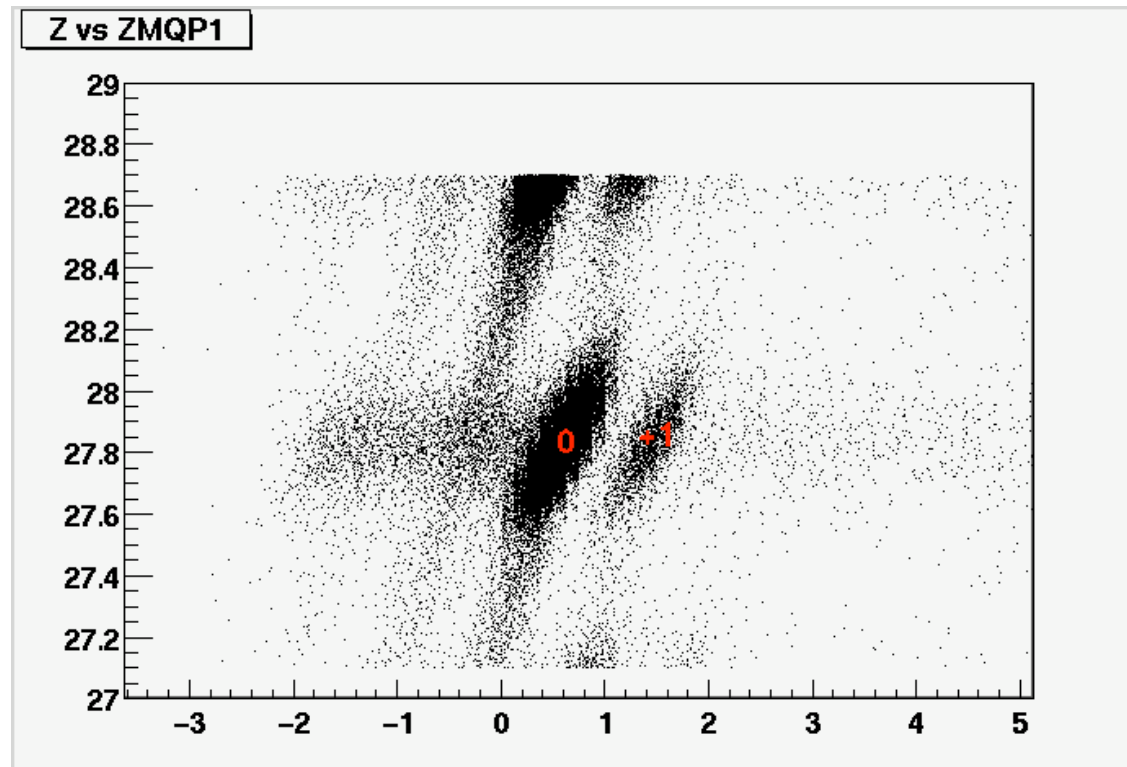
- On va analyser des données issues d'une expérience LISE* qui a pour but de mettre en évidence des différences dans des spectres en énergie des γ pour deux isotopes du nickel. Le fichier de données est **[r50_69ni.root](#)**.
- On va procéder par étapes successives:
 1. sélection du bon état de charge
 2. sélection des deux isotopes de Ni à étudier.
 3. recalage des temps pour pouvoir cumuler des histogrammes.
 4. fabrication des spectres en énergie des γ pour les deux isotopes

http://caeinfo.in2p3.fr/root/Formation/fr/Jour4/r50_69ni.root

**Merci à M.Sawicka, F.De Oliveira et J.M.Daugas!*

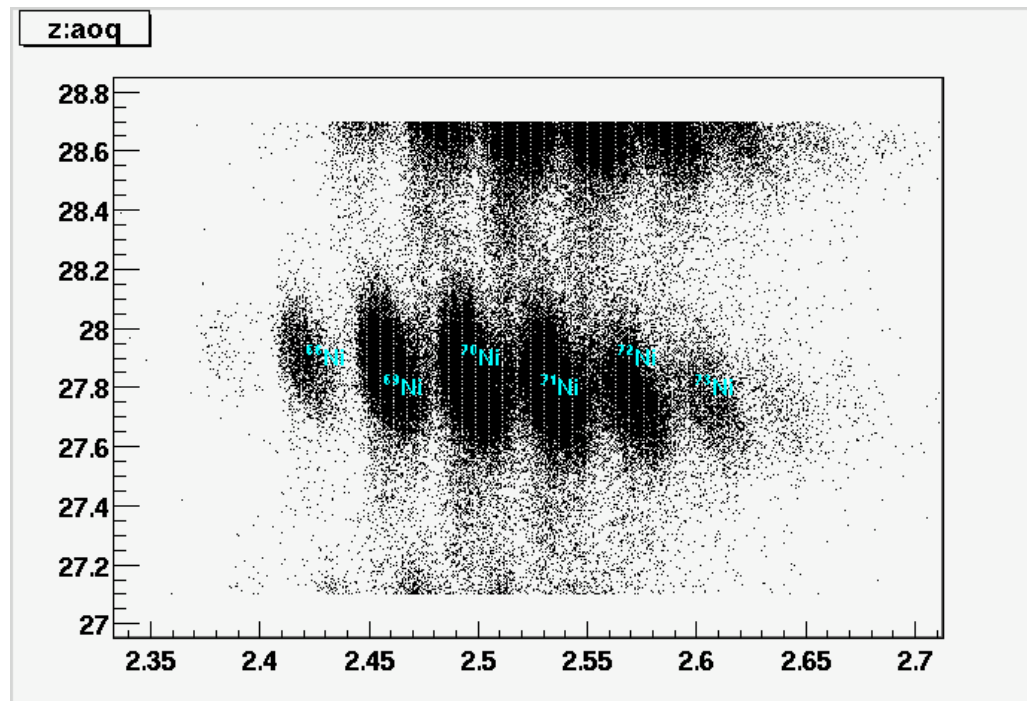
Exercice: étape 1

- Sélection de l'état de charge
 - faire l'histogramme **z** en fonction de **zmqp1**.
 - Faire une coupure graphique nommée CUTEC autour de la bosse centrée en (0,28)



Exercice: étape 2

- Sélection des isotopes du Ni:
 - faire l'histogramme **z** en fonction de **aoq**
 - faire une sélection graphique CUTNI69 autour de la bosse centrée en (2.45,27.9)
 - faire une sélection graphique CUTNI70 autour de la bosse centrée en (2.5,27.9)



Sauvegarder les coupures

```
#include "TFile.h"
#include "TCUTG.h"

void SaveCuts(void)
{
  TFile *fcoup=new TFile("coupures.root","recreate");
  fcoup->cd();
  gROOT->FindObject("CUTEC")->Write();
  gROOT->FindObject("CUTNI69")->Write();
  gROOT->FindObject("CUTNI70")->Write();
  fcoup->Close();
}
```

<http://caeinfo.in2p3.fr/root/Formation/fr/Jour4/HandleCut.C>

Pour retrouver les coupures

```
void LoadCuts(void)
{
TFile *fcoup=new TFile("coupures.root");
TCutG *CUTEK=(TCutG *)fcoup->Get("CUTEK");
TCutG *CUTNI69=(TCutG *)fcoup->Get("CUTNI69");
TCutG *CUTNI70=(TCutG *)fcoup->Get("CUTNI70");
fcoup->Close();
}
```

Utilisation:

`root[0] .L HandleCuts.C+`

`root[1] SaveCuts()`

pour les sauvegarder

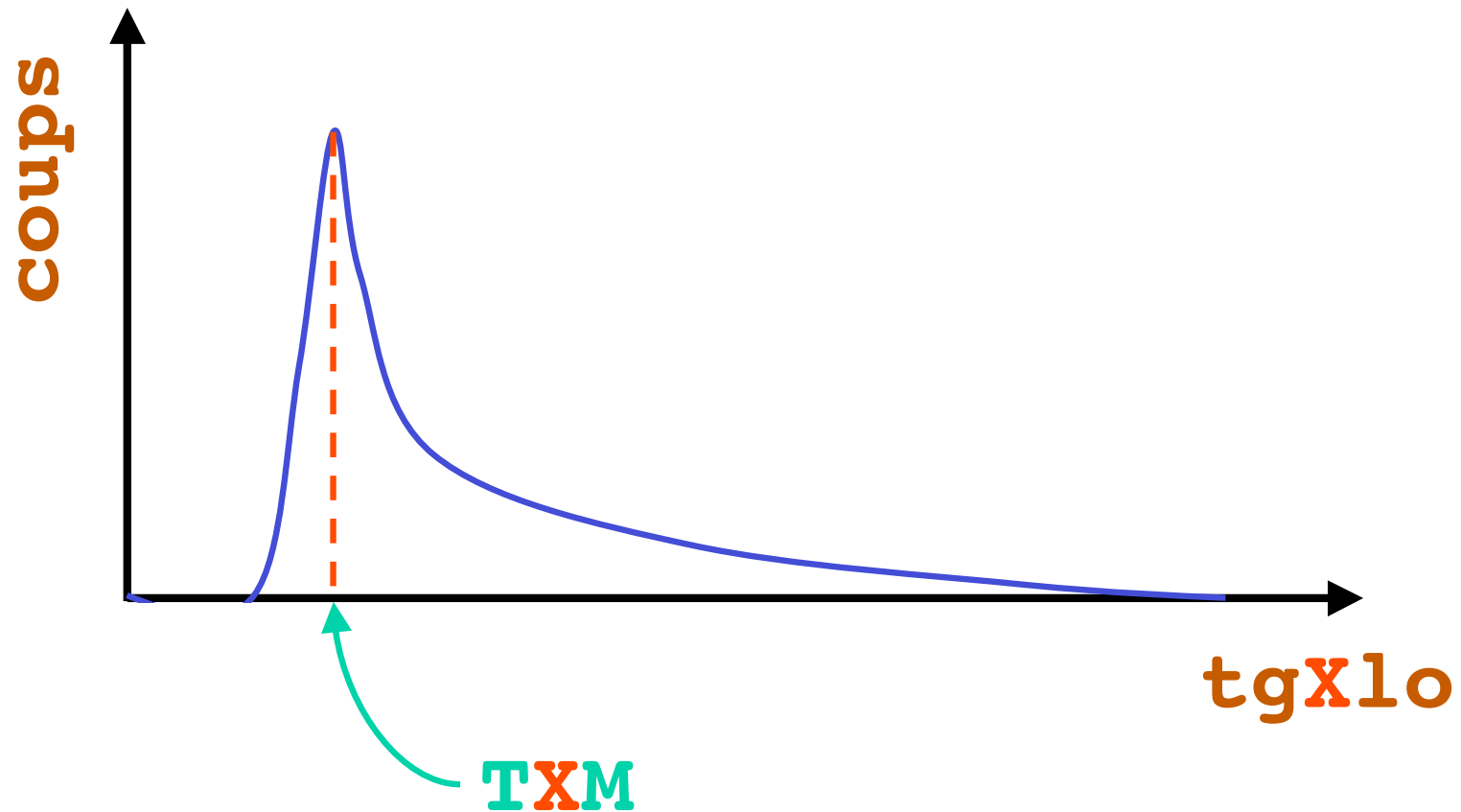
`root[2] LoadCuts()`

pour les lire

Exercice: étape 3

- Recalage des spectres en temps « long ».
 - fabriquer l'histogramme de **tg1lo** pour des valeurs inférieures à 3000
 - repérer l'abscisse **T1M** du maximum de l'histogramme
 - fabriquer l'alias **RTG1LO** = **tg1lo** - **T1M**
 - recommencer ces opérations pour les 5 autres **tgXlo** pour **X**=2 à 6.

Etape 3: une petite explication



Exercice :étape 4

- **Fabriquer les spectres suivants pour les états de charge 0**
 - cumul des spectres **EgXc** pour **X** de 1 à 6
 - même spectre que précédemment pour le ^{69}Ni
 - même spectre que précédemment pour le ^{70}Ni
 - superposer ces histogrammes
 - conclusions?
- **Fabriquer les spectres suivants pour les états de charge 0**
 - cumul des spectres **EgXc** vs RTG**X**LO pour **X** de 1 à 6 pour le ^{70}Ni .
 - Faire les projections de ce spectre sur l'axe des temps pour les 2 pics en énergie les plus intenses ($E_\gamma \approx 183 \text{ keV}$ et $E_\gamma \approx 447 \text{ keV}$)
 - extraire des spectres projetés les temps de demi-vie de ces 2 pics γ (avec un fit d'un fond constant + une exponentielle)
 - conclusions?