Statistiques non-paramétriques : Ch. 3. Économétrie non-paramétrique 2018-19

M2 CEE

Pr. Philippe Polomé, Université Lumière Lyon 2

Sommaire

Densités & histogrammes lissés

- On commence par une analyse non-conditionnelle
  - Une régression est une analyse conditionnelle
  - Ici on veut la courbe des "y"
    - Possiblement y est multidimensionnel
- On va regarder une façon de présenter un histogramme
  - de façon graphique
  - en lissant les bords avec un "kernel smoother"
    - Smooth : lisser
    - Kernel : noyau
    - "Densité par lissage noyau"
    - ou "densité noyau"

Histogramme

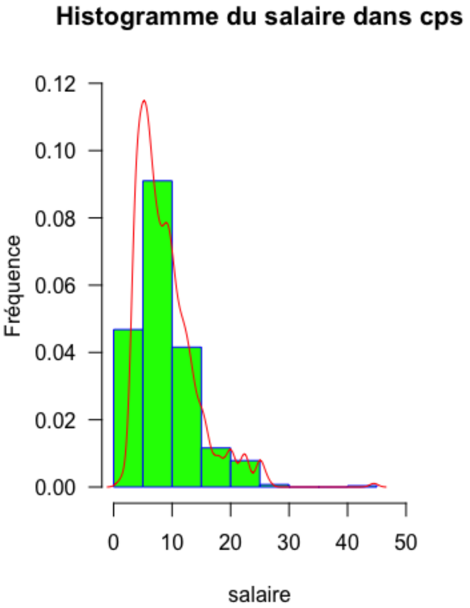
- Un histogramme
  - est une estimation non-param. de la densité $f(x)$ d'une va x
    - que l'on forme en divisant le **support** de x en intervalles **également espacés**
    - et en calculant la fraction de l'échantillon dans chaque intervalle
- Dans R
  - Données DataFrame cps
    - Dans package AER
  - Si vous utilisez le projet "Cours R" du cours de programmation
    - vous avez le fichier cps qui est chargé (sur les salaires)
  - Fonction hist( )
    - est un basique de R
    - pas besoin de charger un package
  - hist(cps$wage)

Histogramme dans R : personnalisations

```
hist(cps$wage, main="Histogramme du salaire dans cps",
xlab="salaire", border="blue", col="green", xlim=c(0,50), las=1,
breaks=10)
```

- breaks est le nombre d'intervalles
  - Peut changer beaucoup l'aspect
    - On y revient + loin
- Mettre les noms d'axes & de couleurs entre " "
- las sert a changer la présentation des labels sur Y (0,1,2,3)
- On peut présenter des freq. rel. avec "freq = F"
  - Plutôt que des fréq. abs.
- On peut ajouter une densité lisse par dessus avec
  - lines(density(cps$wage))
    - Cette densité lisse est prédéfinie
    - On va explorer des méthodes pour la tracer

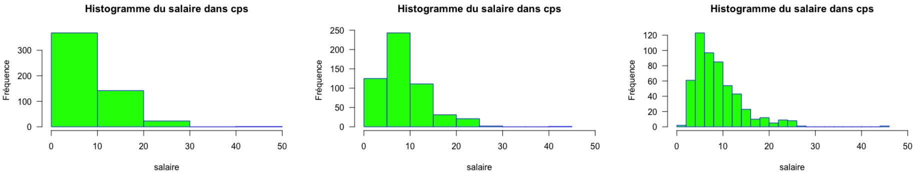
Histogramme dans R → np2017.r



Effet de la bandwidth dans un histogramme → np2017.r

hist(cps$wage, breaks=10)

- breaks est le nombre d'intervalles
  - Donc de leur largeur, appelé **bandwidth**



- Exercice
  - Ajuster le nombre d'intervalles
  - Changer la couleur (rcolor.pdf sur google)

Estimateur Histogramme

- Plus formellement, dans un histogramme,
  - on veut estimer la densité $f(x_0)$ d'une v.a. scalaire continue x
    - évaluée en x_0
- Si on a un échantillon $\{x_i, i = 1, ..., N\}$,
  - **l'estimateur histogramme** de $f(x_0)$ est

$$\hat{f}_{\mathtt{hist}}(x_0) = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathbf{1}(x_0 - h < x_i < x_0 + h)}{2h}$$

  - $2h$ est la longueur de l'intervalle
    - base du rectangle
  - $\mathbf{1}(A)$ est une fonction indicatrice$= 1$ si A arrive et 0 sinon
    - Donc : on compte le nbr d'obs. **autour** de x_0 dans un rayon h
    - C'est un estimateur **local** car il n'utilise que de l'info locale

Densité noyau

- Cette procédure amène à une **estimation de la densité** qui est en escalier
  - Même si la véritable densité est lisse
- On réécrit l'estimateur Histogramme comme

$$\hat{f}_{\texttt{hist}}(x_0) = \frac{1}{Nh}\sum_{i=1}^{N}\frac{1}{2}\mathbf{1}\left(\frac{|x_i - x_0|}{h} < 1\right)$$

- L'estimateur **densité noyau** DN généralise cette définition
  - en remplaçant la fonction $\mathbf{1}(.)$ par une alternative $K(.)$

$$\hat{f}_{NOYAU}(x_0) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{x_i - x_0}{h}\right)$$

  - $K(\cdot)$ est dite "fonction noyau" (kernel)
    - un "kernel" est simplement une fonction de pondération
  - h est appelé largeur de bande
    - Paramètre de lissage ou **bandwidth**

Densité noyau

- Un estimateur noyau d'une densité
  - est donc une version lisse de son **histogramme**
  - évaluée **en chaque point** de l'échantillon
    - au lieu de quelques points comme dans l'histogramme
- Il s'agit d'un **estimateur** de la fonction de densité
  - souvent appelé Rosenblatt–Parzen
    - Rosenblatt (1956), Parzen (1962)

Densité noyau

- La fonction noyau K est positive, intégrable et à valeurs réelles
  - Souvent sym autour de 0, on note $z = \frac{x - x_0}{h}$
    - L'uniforme $\frac{1}{2}\mathbf{1}\left(|z| < 1\right)$ correspond à l'histogramme
    - Quadratique $\frac{3}{4}\left(1 - z^2\right)\mathbf{1}\left(|z| < 1\right)$
    - Gaussienne$(0, 1)$: $(2\pi)^{-1/2}\, exp\left(-z^2/2\right)$
    - Gaussienne$\left(\mu, \sigma^2\right)$: $(2\pi)^{-1/2}\, exp\left(-\left(\frac{z-\mu}{\sigma}\right)^2/2\right)$
  - Le choix (arbitraire) du noyau est réputé comme peu influent sur l'estimateur
- h, le paramètre de lissage, est plus difficile à choisir
  - Le + petit le + lisse
    - Mais trop petit, il provoque l'apparition de détails artificiels sur le graphe
    - car trop peu de données entrent dans l'intervalle
  - Trop grand, la majorité du relief est effacée

Kernel smoother : Exemple density

- density(x, bw = "nrd0", kernel, n = 512)
  - x : vecteur des données
  - bw : le choix de la bandwidth
    - Pls "rules of thumb" : SJ based on derivatives, nrd0 (défaut, mais pour raisons de compatibilité), ucv (unbiased cross-validation), bcv (biased cross-validation)
    - On verra + loin la cross-validation
  - kernel =
    - "gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine"
  - n : nbr des points equidistants auxquels estimer le kernel "gridsize"
    - On pourrait estimer en chaque point de l'éch.
- On va utiliser plot pour comparer les options de density

Kernel smoother : Autre exemple bkde

- Charger/installer le package KernSmooth
  - Comme dans le cours de programmation
  - library("KernSmooth")
- wage_bkde <- bkde(cps$wage, kernel = "normal", bandwidth=1)
  - bkde : Binned Kernel Density Estimate
  - Utiliser la variable “wage” dans le DataFrame cps
  - Kernel (noyau) : normal (arbitraire)
    - Autres : box, epanech, biweight, triweight
  - Bandwidth : 1 (arbitraire)
    - défaut : un calcul à partir de la variance de x
- lines(wage_bkde)
  - On voit que bkde est comme density
  - plot pour comparer les options de bkde & de density

plot(bkde(cps$wage, kernel = "normal", bandwidth=1))

- Dans les 2 cas, density & bkde
  - On voit que le bandwidth impacte beaucoup
  - Par contre, le kernel lui-même n'impacte pas beaucoup
  - La gridsize ne change que la question du dessin et n'impacte pas la forme de la courbe
- Quelques options de présentation de plot
  - plot(wage_bkde, col=rev(rainbow(400, s = 1, v = 1, start = 0, end = max(1,400 - 1)/400, alpha = 1)), xlab = "wage", ylab="density")
    - La couleur reflète la valeur de wage pas celle de density
- Diapo suivante : généraliser à deux dimensions
  - Cfr cours Prog dans R

Kernel smoother : Exemple 2D → np2017.R

- Densité des observations en bivarié
  - **bivariate** binned kernel density estimator bkde2D( )
- cps_bkde <- bkde2D(cbind(cps$experience, log(cps$wage)), bandwidth=c(3.5, 0.5), gridsize=c(200, 200))
  - Il faut choisir la bandwidth et la taille de la grille sur chacune des 2 dimensions
    - expérience & log(wage)
  - gridsize en 2D
- image(cps_bkde$x1, cps_bkde$x2, cps_bkde$fhat, col=rev(gray.colors(10, gamma=1)), xlab = "experience", ylab="log(wage)")
  - "heatmap" / courbes de niveau
- Exercice
  - Récupérer les données CPS1988 du package AER
    - Essayer de trouver la meilleure représentation en 1D et 2D
    - En changeant la bandwidth et/ou le kernel

Interpréter les densités

- Sur le fond, la densité noyau (univariée) est une ligne
  - que l'on dessine
  - qui s'ajuste d'une certaine manière au nuage de points
    - concrètement : elle représente la fréquence locale en chaque point
- On est proche d'une logique de régression
  - Il faut conditionner
  - Il y a d'autres façons de dessiner
    - Splines, "nearest neighbor", "neural network"...
  - Dans ce cours, on reste sur le noyau.

Sommaire

Propriétés de l'estimateur noyau univarié

- En 1D, 2D ou plus,
  - La fonction kernel doit se comporter comme une densité
    - de moyenne nulle
    - et de variance finie

$$\int K(z)\,dz = 1 \quad \int zK(z)\,dz = 0 \quad \int z^2 K(z)\,dz = \kappa_2 < \infty$$

  - Le support est généralement $-\infty, +\infty$

Erreur carrée moyenne (mean square error) MSE

- Les fonctions noyau sont svt choisies sur un critère de MSE
- Le bias de l'estimateur $\hat{f}_{NOYAU}(x)$ est

$$\hat{f}(x) - f(x)$$

(on laisse tomber "NOYAU" quand il n'y a pas confusion)

- La MSE est

$$mse\hat{f}(x) = E\left(biais^2\right) = var\,\hat{f}(x) + biais^2$$

on peut montrer que

$$biais\,\hat{f}(x) \approx \frac{h^2}{2}\frac{\partial f(x)}{\partial x^2}\kappa_2$$

$$var\,\hat{f}(x) \approx \frac{f(x)}{nh}\int K^2(z)\,dz$$

Erreur carrée moyenne (mean square error) MSE

- On voit que
  - La variance **diminue** avec la bandwidth h
  - Le biais **augmente** avec le carré de la bandwidth h
  - Le biais augmente avec la dérivée 2º de $f\,()$
    - il est donc au + fort autour du/des pics de la distribution
- Ces formules ont lieu en un point
- On peut intégrer le mse sur z pour obtenir une mse globale

$$imse\,\hat{f} = \int_{support\,x} mse\,\hat{f}\,(x)\,dx$$

- On cherche le kernel K et la bandwidth h qui minimisent $imse$

Erreur carrée moyenne (mean square error) MSE

- Le Kernel optimal est

$$K_e\left(z\right) = \begin{cases} l@ \quad l@@l@ \quad l@ \ \frac{3}{4\sqrt{5}}\left(1 - \frac{1}{5}z^2\right) & -\sqrt{5} \leq z \leq \sqrt{5} \\ 0 & sinon \end{cases}$$

  qu'on appelle kernel d'Epanechnikov du nom de son inventeur
- Il se fait que pas mal de kernels ont des propriétés semblables
  - donc le kernel est souvent choisi pour des raisons informatiques
    - le kernel gaussien est le + souvent choisi

Choix de bandwidth

- Au contraire du Kernel,
  - l'optimisation précédente n'amène pas à une bandwidth utilisable en pratique
  - Mais la bandwidth détermine l'ajustement bien plus que le kernel
- Il est important d'en choisir une appropriée au problème traité
  - Il y a 4 grandes approches

1. Heuristique (rule of thumb)
2. Plug-in – je ne poursuis pas
3. Validation croisée
   3.1 par Moindres Carrés
   3.2 par Maximum de Vraisemblance
4. Bootstrap

Heuristique “référence”

- Le principe est d’utiliser le kernel choisi pour calculer la bandwidth optimale
- Comme souvent ce kernel est le kernel gaussien
  - Ça donne une bandwidth de $1.06\sigma n^{-1/5}$
    - n est la taille d’éch.
    - σ est l’écart-type de la normale utilisée dans le kernel
    - en pratique, on prend $\hat{\sigma}$ l’écart-type de l’échantillon

Autres méthodes

- Validation croisée par Moindres Carrés
  - Il est possible d'estimer le imse à partir de l'éch.
  - Cet imse dépend du bandwidth
  - On peut alors choisir le bandwidth qui minimise l'imse
  - Cette approche est la meilleure de celles présentées
    - mais est sensible à de petites variations des données (arrondis...)
- Validation croisée par Maximum de Vraisemblance
  - Même idée que la précédente, autre façon d'estimer le imse
  - Tend à sur-lisser (gommer les variations)
- Par bootstrap
  - Toujours l'idée d'estimer le imse
  - Trop exigeante sur le plan calculatoire

Conclusion

- On pourrait poursuivre avec
  - Comment estimer une densité discrète
  - Comment estimer une multivariée
    - Illustration graphique prochaine dia
- On va plutôt passer à l'analyse conditionnelle
  - Prélude à la régression
- Sur l'estimation d'une densité inconditionnelle
  - La comparaison n'est pas tellement "paramétrique" vs. np
    - car paramétrique est souvent mal spécifié, donc inconsistant
  - alors que np ne peut pas être mal spécifié
    - mais par contre est assez inefficient

Estimations de densités multivariées par np dans R

- #### dynamic bivariate density plot avec donnees simulees
  - Exécuter le programme
    - Sélectionner tout le progr
    - jusque #### FIN dynamic bivariate density plot simulation normales
    - et "run"
    - Mettez en grand la fenêtre de sortie (zoom ou agrandir manuellement)
  - 'sliders' and 'pickers' qui permettent de changer
    - kernel function & order (une propriété math du kernel qu'on ne voit pas)
    - scale factors (la bandwidth est calculée automatiquement, scale factor prend une part de cette bandwidth, donc équivalent à la bandwidth)
    - azimuthal viewing direction (point de vue)
    - number of training (essentiellement = gridsize)
    - number of evaluation observations (taille de l'éch.)
- #### dynamic bivariate density plot avec donnees réelles Geyser
  - Idem sur données réelles

Sommaire

Estimation d'une densité conditionnelle

- La densité conditionnelle est sous-jacente à l'analyse de régression
  - Mais est rarement modélisée directement
  - En np, il est plus clair de passer par cette étape
- Soit $f(.)$ la densité jointe de (X, Y)
  - pour rappel $\Pr\{X \leq x, Y \leq y\} = \int^x \int^y f(.)\, dydx = f(x, y)$
  - $\mu(.)$ la densité marginale de X
    - $\mu(.) = \int f(.)\, dY$
    - "integrate Y out"
- Pour la suite,
  - Y est la variable dépendante
    - est expliquée
  - X est un régresseur
    - est explicative

Densité conditionnelle

- La densité conditionnelle est

$$\Pr\{Y \leq y|X \leq x\} = g(y|x) = f(x,y)/\mu(x)$$

- L'**estimateur noyau** de cette densité est

$$\hat{g}(y|x) = \hat{f}(x,y)/\hat{\mu}(x)$$

  - $\hat{\mu}(x)$ est l'estimateur noyau univarié vu à la section précédente
  - $\hat{f}(x,y)$ est une extension bivariée de cet estimateur
  - Il faudrait discuter le choix de bandwidth
  - La commande correspondante de np est npcdensbw
    - Conditionnal DENSity avec BandWidth par cv.ls (cross-validation par Least squares)
    - En général, on n'utilise pas directement cette commande, qui est appelée par d'autre procédures

R #### Least-squares cross-validated conditional density estimation

- Génération de données (normale bivariée)
  - n <- 500
  - rho <- 0.25 pour sigma ci-dessous
  - mu <- c(0,0) moyenne des variables a generer
  - Sigma <- matrix(c(1,rho,rho,1),2,2) matrice de var-cov des variables a générer
    - donc ici : générer un normale bivariée dont le mu contient 2 moyennes nulles (dist. marg.) et sigma indique une matrice avec des variance a 1 et des cov a rho=.25
    - rem. les deux chiffres qui suivent dans sigma indiquent "2L 2C"
  - data <- mvrnorm(n=n, mu, Sigma) génère des valeurs d'une normale multivariée
  - mydat <- data.frame(x=data[,2],y=data[,1])
    - y et x dont la corrélation est contrôlée par rho

Estimation

- npcdensbw va estimer la bandwidth associée à $\hat{g}(y|x) = \hat{f}(x, y) / \hat{\mu}(x)$
  - On verra + précisément comment + loin
  - Prend un peu de temps, selon la machine
  - Output summary assez sommaire (juste bandwidth)
  - Bandwith type : fixed = kernel, il y en a d'autres
- Output graphique
  - Sélectionner la fenêtre plot
  - (x, y, densité) : on doit voir un graphique qui tourne
- Exercice : Pour voir comment la relation entre x et y change, changer les paramètres de la normale bivariée
  - corrélation rho élevée, $\oplus$ ou $\ominus$
    - moyennes mu non-nulles
    - variances fort différentes (à l'intérieur de sigma)

Régresseurs non-pertinents

- Pour estimer une fonction $g(y|x) = f(x,y)/\mu(x)$
  - conditionnelle à pls régresseurs x vecteur
  - il faut intégrer autant de fois que de régresseurs
    - pour obtenir la distribution multidim. des régresseurs
- On peut montrer que la convergence de $\hat{f}(.)$ à $f(.)$
  - se détériore rapidement lorsque le nombre de variables continue augmente
  - "**malédiction de la dimensionalité**"
- Il est donc particulièrement important en np
  - d'éviter les régresseurs non-pertinents
  - Idéalement, ils sont "smoothed out" :
    - Pour un x non pertinent, le graphe de y reste le même pour tous les niveaux de x

Régresseurs non-pertinents

- Hall et al. (2004) montre que
  - Une version de la validation croisée par MC
    - assigne automatiquement un fort paramètre de lissage aux régresseurs non-pertinents
    - leur dist. marginale tend à l'uniforme
    - cela supprime leur contribution à la variance de l'estimateur
    - et donc montre qu'ils sont indép. de la variable expliquée
  - Les variables pertinentes par contre sont lissées de façon usuelle
    - Pas vers l'uniforme
- Donc choisir la validation croisée par MC
  - permet de trier les régresseurs pertinents et non.

Sommaire

Sommaire

Cas bivarié

- Cet estimateur de régression est connu aussi sous le nom "Nadaraya–Watson"
  - de ceux qui l'ont proposé
- On ne prend qu'un régresseur pour commencer
  - par simplicité de notation

Moyenne conditionnelle $\hat{g}(x)$

- Par définition, la moyenne conditionnelle de Y continue est

$$g(x) = \int y g(y|x)\, dy = \int y \frac{f(y,x)}{f(x)} dy = \frac{m(x)}{f(x)}$$

  - $g(y|x)$ est la densité conditionnelle de la section précédente
  - $m(x) = \int y f(y,x)\, dy$ une moyenne "partielle"
- L'estimateur Kernel Local Constant KLC
  - est celui défini à la section précédente

$$\hat{g}(x) = \int y \frac{\hat{f}(y,x)}{\hat{f}(x)} dy = \frac{\sum_{i=1}^{n} y_i K\left(\frac{x_i - x}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{x_i - x}{h}\right)}$$

  - c'est un estimateur consistant de la moyenne conditionnelle

Biais et variance

- L'estimateur Kernel Local Constant KLC
  - souffre de "biais sur les bords"
  - p.e. on peut calculer dans le cas bivarié que

$$\text{biais} \approx h^2 \left(\frac{1}{2} g^{''}(x) + \frac{g^{'}(x) f^{'}(x)}{f(x)} \right) \kappa_2$$

  - Quand on approche du "bord" des données
    - $f(x) \to 0$ donc le biais augmente
    - Ce problème n'est pas partagé par l'estimateur "polynome local" de la section suivante car le 2º terme à l'intérieur de la parenthèse disparait dans le cas linéaire

Bandwidths : optimale et basée sur données

- La bandwidth optimale de l'estimateur KLC
  - dépend de quantités inconnues, comme précédemment
  - Elle ne peut être calculée dans le cas de régression
  - On va donc utiliser une bandwidth basée sur les données
- Deux calculs de bandwidth basés sur les données
  - Sont populaires
  - Validation croisée moindres carrés (cfr section précédente)
  - Minimiser le critère d'information d'Akaike (Hurvich)
  - On a montré qu'ils sont asymptotiquement équivalents

Régresseurs pertinents et non-pertinents

- Il a été montré (cfr sect. densité cond.) que
  - la validation croisée par MC
  - mène à un lissage optimal des 2 types de régresseurs
    - Les non-pertinents n'ont plus d'effet sur la variance de l'estimateur
- La malédiction de la dimensionalité
  - implique que les non-pertinents doivent être retirés de la régression
  - afin de réduire le bruit autour des pertinents

Effets marginaux avec KLC "$\hat{\beta}(x)$"

- On appelle "effet marginal" ou "réponse"
  - L'effet de x sur $g(x) = \int yg(y|x)\,dy$
    - donc sur la moyenne conditionnelle de y
  - Par analogie avec le modèle de régression linéaire $y = x\beta + \epsilon$
    - on appelle cette réponse $\beta(x)$
- $$\beta(x) = \frac{\partial g(x)}{\partial x} = \frac{f(x)\,m^{'}(x) - m(x)\,f^{'}(x)}{f^{2}(x)}$$
  - puisque $g(x) = \dfrac{m(x)}{f(x)}$
  - donc $\beta(x) = \dfrac{m^{'}(x)}{f(x)} - g(x)\dfrac{f^{'}(x)}{f(x)}$
- On remplace par les estimations de ces valeurs

Effets marginaux avec KLC "$\hat{\beta}(x)$"

$$\hat{\beta}(x) = \frac{\hat{m}'(x)}{\hat{f}(x)} - \hat{g}(x)\frac{\hat{f}'(x)}{\hat{f}(x)} \text{ avec}$$

$\hat{m}(x) = \frac{1}{nh}\sum_{i=1}^{n} y_i K\left(\frac{x_i - x}{h}\right)$	$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x_i - x}{h}\right)$
$\hat{m}'(x) = -\frac{1}{nh^2}\sum_{i=1}^{n} y_i K'\left(\frac{x_i - x}{h}\right)$	$\hat{f}'(x) = -\frac{1}{nh^2}\sum_{i=1}^{n} K'\left(\frac{x_i - x}{h}\right)$

- L'effet marginal n'est jamais constant $\leftrightarrow$ régression linéaire
  - On verra mieux dans les exemples
- En multivarié (plusieurs régresseurs)
  - C'est plus compliqué, mais le fond est le même

Conclusion sur Kernel Local Constant

- On a donc un estimateur de **régression** np
  - à Validation croisée moindres carrés
  - qui fait que
    - les régresseurs non-pertinents disparaissent d'eux mêmes
    - on peut calculer les effets marginaux $\hat{\beta}(x)$
- On va voir un autre estimateur

Sommaire

Analogie

- Le KLC peut être réécrit comme minimisation de

$$\hat{g}(x) \equiv \min_{a} \sum_{i=1}^{n} (y_i - a) K\left(\frac{x_i - x}{h}\right)$$

- On généralise en mettant un polynome à la place de a
  - Le plus populaire est le linéaire

$$\hat{g}(x) \equiv \min_{a,b} \sum_{i=1}^{n} (y_i - a - b(x_i - x))^2 K\left(\frac{x_i - x}{h}\right)$$

    qui s'interprête en disant qu'autour d'un point x_0, la régression est approx. linéaire

$$g(x_0) \approx a + b(x_0 - x)$$

  - C'est l'estimateur Kernel Local Polynomial (ou Linéaire)
    - KLP ou KLL

Propriétés

- Le terme a est alors la moyenne conditionnelle $g(x)$
  - Comme l'intercept dans une régression linéaire
- Le terme b est la pente,
  - donc l'effet d'un changement marginal de x sur $g(x)$
  - c'est-à-dire le $\hat{\beta}(x)$ de l'estimateur KLC
- Cet estimateur KLP souffre moins du "biais de bord" que KLC
  - mais est sujet à des problème de singularité
    - $=$ n'est pas défini
    - lorsqu'il y a localement peu de données
- On peut calculer des biais et variances approximés
  - comme avec KLC
- Les régresseurs non-pertinents
  - ne disparaissent pas d'eux-mêmes
    - ne sont pas "smoothed-out" comme avec KLC
  - provoquent une variabilité excessive

Sommaire

Ex 1. Régression sur données simulées #### Regr. sim.

- Génération des données
  - n <- 50 Ech de taille 50
  - x <- sort(runif(n)) runif : suffixe r pour donnees sim, unif pour uniforme
  - x.seq <- seq(0,1,length=1000) séquence de 1000 chiffres de 0 à 1, servira à faire des graphes
  - y <- sin(2*pi*x) + rnorm(n,sd=0.25) valeurs simulées de y

Ex 1. Régression sur données simulées #### Regr. sim.

- Régressions np
  - Calcul des bandwidths
    - bw.lc <- npregbw(y~x,regtype="lc") y~x, kernel local constant
    - bw.ll <- npregbw(y~x,regtype="ll") idem, kernel local linéaire
  - Commande npreg
    - model.lc <- npreg(bws=bw.lc,gradient=TRUE) il faut donner la bandwidth
    - les points de données sont appeles "training points"
    - gradient = true indique que les gradients (donc les pentes, les betas) sont stockes dans un objet "npregression"
    - summary(model.lc) la sortie, assez fruste, pas de coef estimé par définition

Ex 1. Régression sur données simulées

- Premières sorties graphiques
  - plot(x,y,cex=0.2) les données
  - lines(x.seq,sin(2*pi*x.seq),col="black",lty=1) les y sans erreur (bruit)
  - lines(x,sin(2*pi*x),col="red",lty=1) idem mais sur les n=50 points de l'ech.
  - lines(x,fitted(model.lc),col="red",lty=2) y ajustes avec lc, fitted(model.ll)
    - $\hat{g}(x)$ = estimation de l'espérance cond. de y|x
    - "biais de bord"
  - lines(x,fitted(model.ll),col="blue",lty=3) y ajustes avec ll - assez proche
  - pls comparaisons avec moindres carrés
- Exercice. Répétez le programme en changeant
  - la taille d'éch. à 100
  - la façon de générer y
    - p.e. comme la somme ou le ratio de 2 normales pour avoir une dist. bimodale

Ex 1. Régression sur données simulées

- Deuxième sortie graphique
  - gradient $\hat{\beta}(x) = \dfrac{\partial \hat{g}(x)}{\partial x}$ en fonction de x : non-linéaire
  - plot(x,2*pi*cos(2*pi*x),ylab="dy/dx",col="black",type="l",lty=1) $\beta(x)$ "réel" dy/dx
  - lines(x,gradients(model.lc)[,1],col="red",lty=2) $\hat{\beta}(x)$ estimé par lc
  - lines(x,gradients(model.ll)[,1],col="blue",lty=3) $\hat{\beta}(x)$ estimé par ll
- Exercice : gradient MCO du MRL bien spécifié lm(y~sin(2*pi*x)) et du quadratique lm(y~x+I(x^2))

Ex 1. Régression sur données simulées

- Troisième sortie graphique : Effet de la bandwidth
  - bw.lc$bw[1] <- 1e05 Change la bandwidth : bcp trop grande
  - bw.ll$bw[1] <- 1e05
  - model.lc <- npreg(bws=bw.lc,gradient=TRUE) Recalculer les régressions à p. de ces bandwidths
  - model.ll <- npreg(bws=bw.ll,gradient=TRUE)
  - Graphique
    - plot(x,y,cex=0.2) Les données
    - lines(x.seq,sin(2*pi*x.seq),col="black",lty=1) les y sans erreurs
    - lines(x,fitted(model.lc),col="red",lty=2) valeurs ajustées : droite plate
    - lines(x,fitted(model.ll),col="blue",lty=3) valeurs ajustées : droite pente nég.
  - On voit pourquoi on appelle aussi les régressions np "scatterplot smoothing"
    - Elles aident à voir les relations entre variables

Ex. Comparaison de méthodes de choix de Bandwidth

- #### 3.2.3.b Regr. bandwidth (data Prestige du pack. car)
- 5 bandwidths
  - 1 plug-in du package KernSmooth
    - Avec la commande locpoly de régression np de ce package
  - Pour les 4 autres, commande npregbw du package np
    - servant à optimiser la bandwidth
  - 2 arbitraires à partir du package np sans optimiser, avec Local Linear
    - On prend un chiffre en lien avec la mesure du régresseur
    - On divise : "undersmoothed bandwidth" trop fine → ajustement trop collé aux données
    - On multiplie "oversmoothed badnwidth" trop large → faible ajustement
  - 2 optimales selon package np, avec cross-validation : Moindres carrés et AIC
    - Ce sont les 2 cross-validations fournies par le package
- Plot en 4 tableaux
  - Regroupe les 2 cv

Ex. Comparaison de méthodes de choix de Bandwidth

- Exercice
  - Séparer le plot des 2 cv en 2 et supprimer le plug-in
  - Changer les 2 bw arbitraires under et over pour les rapprocher
  - Comparer avec lc au lieu de ll

Et le t ?

- Comment voit-on la significativité de l'effet marginal ?
- Les graphes mettent des intervalles de confiance
  - On va voir ça dans le prochain exemple
- La section suivante “Test d'hypothèse consistant”
  - présente des tests formels (non graphiques)

Ex. Multivariate Mixed-Data Application

- #### 3.2.3.c Ex. regr. mixte (cont. & cat) Part 1
  - D'abord un graphique exploratoire
  - Graphique en 3D et en perspective
    - npudensbw(~lwage+ordered(numdep),data=wage1) calcule la bandwidth pour une bivariee (les 2 var sont a D du ~)
    - Pq ordered ? pcq + loin on utilise pour le graphique, il faut que les natures des variables restent les mêmes

Ex. Multivariate Mixed-Data Application

- #### 3.2.3.c Ex. regr. mixte (cont. & cat) Part 2
  - On calcule une bandwidth pour la régression de lwage sur female, married, educ, exper, tenure
    - Pas besoin de exper^2 c'est la régression qui regarde la relation
    - La bandwidth diffère par régresseur
  - Graphique de la moyenne conditionnelle de y à chaque niveau du régresseur
    - On voit les effets "classiques" : salaire plus bas pour les femmes, rendement décroissant de l'exp. (plus prononcé que d'habitude à cause de tenure sans doute)
    - Constater graphiquement que la significativité n'est pas constante puisque l'intervalle de confiance évolue selon x
  - Graphique du gradient
    - Évolution du "coefficient" $\hat{\beta}(x)$ en fonction du x
    - Avec son intervalle de confiance graphique

Exercice

- Répliquer en changeant la CV ou le kernel (LC au lieu de LL)
- Répliquer avec les données bwages du package Ecdat

Sommaire

Intro

- On ne revient pas sur les tests np de la 1º partie du cours
  - il s'agit de tests en contexte de régression
- Soit tester une spécification paramétrique
  - Permet de justifier une approche np
- Soit tester la significativité des régresseurs d'une régression np
  - l'équivalent des t-stats
    - Un régresseur parmi pls
  - On n'a pas d'équivalent au test F

Un test de significativité pour des régressions np

- Il y a pls approches
  - On prend ici celle de Racine qui accepte des régresseurs continus & cat.
- On va regarder d'abord cat.
  - puis continu

Un test de significativité : régresseurs catégoriques

- Soit
  - z le régr. cat qui peut être non-pertinent
  - X tous les autres régresseurs
  - L'hyp. nulle est

    $$H_0 : E\left(Y|X,z\right) = E\left(Y|X\right)$$

    presque partout
    L'alternative est que l'égalité est en fait $\neq$ (2-tailed)
- Pour simplifier on écrit
  - $g\left(x\right) = E\left(Y|x\right)$ et $m\left(x,z\right) = E\left(Y|X,z\right)$
  - disons que z prend c valeurs dont la 1º est zéro
    - si $c = 2$, z est une dichotomique, le cas le + fréquent
  - H_0 peut alors s'écrire $m\left(x, z = l\right) = m\left(x, z = 0\right)\ \forall l$

Un test de significativité : régresseurs catégoriques

- La stat de test est un estimateur de

$$I = \sum_{l=1}^{c-1} E\left\{\left[m\left(x, z=l\right) - m\left(x, z=0\right)\right]^2\right\}$$

- Pour la calculer
  - On prend les valeurs estimées par KLC ou KLL de m
  - On somme sur toutes les $l \neq 0$
    - et sur toutes les obs.
  - On voit bien que $I \geq 0$
    - et que z est non signif. si I est proche de zéro
- Il n'y a pas de distribution connue
  - Il faut faire du bootstrap
  - Mis en oeuvre dans npsigtest( )
    - On verra dans l'exemple

Un test de significativité : régresseurs continus

- H_0 est la même,
  - mais "presque partout"
    - car on a en quelque sorte ∞ catégories
  - Équivalent à $\dfrac{\partial E\left(y|x,z\right)}{\partial z}=\beta\left(z\right)=0$ presque partout
- La stat de test est un estimateur de $I=E\left\{\left[\beta\left(z\right)\right]^2\right\}$
  - On calcule son estimation KLC ou KLL $I_n=\frac{1}{n}\sum_{i=1}^{n}\hat{\beta}\left(z_i\right)^2$
  - Comme dans le cas catégorique, la distribution de I_n est inconnue
    - On utilise bootstrap

Ex. Regression mixte (cont & cat.)

- On poursuit l'exemple précédent – part 3
  - #### 3.2.3.c Ex. regr. mixte (cont. & cat)
    - # Part 3 Tests de significativite
  - On avait estimé
    - bw.all <- npregbw(formula=lwage~factor(female)+ factor(married)+ educ+ exper+ tenure, regtype="ll", bwmethod="cv.aic", data=wage1)
  - En pratique, le test est simple
    - on passe l'objet bandwidth de la régression associée
    - Par contre le temps de calcul est plus long
- Le package
  - distingue le cat et le cont
  - fait le test adéquat
- Pas de test sur plusieurs coef conjointement

Tester une spécification paramétrique

- On veut tester si un modèle **param**. est correct

  $$H_0 : E(Y|x) = m(x, \gamma_0)$$

  pour presque tous les x
  pour un certain γ_0 vecteur $(p \times 1)$ de paramètres
  et $m(x, \gamma)$ une fonction connue (possiblement non-linéaire)
- En définissant

  $$\mu_i = y_i - m(x_i, \gamma_0)$$

  alors on peut écrire

  $$H_o : E(\mu_i|x_i) = 0$$

  pour presque toutes les valeurs possibles de x
  - Fondamentalement : les résidus μ_i ne devraient pas dépendre des x_i

Tester une spécification paramétrique

- Un test consistant de spécification du modèle paramétrique peut être construit
  - sur la base des résidus du modèle paramétrique $\hat{\mu}_i = y_i - m(x_i, \hat{\gamma})$
  - en estimant $E(\mu_i | x_i)$ de manière non-paramétrique
  - au moyen d'une technique de bootstrap
- La stat calculée se nomme J_n
  - Mis en oeuvre dans npcmstest( )
  - Il faut d'abord estimer soit un lm soit un glm
    - le y doit être continu, donc pas probit, logit, Poisson
    - en précisant les arguments x=TRUE, y=TRUE qui font que x et y vont être stockés dans l'objet résultat
    - Ensuite il faut stocker les x dans un dataframe
    - npcmstest prend comme arg l'objet résultat du modèle param., le x et le y
  - On peut employer summary sur npcmstest
    - Si on R H0, le modèle param est mal spécifié

Ex. Tester une spécification paramétrique

- On poursuit l'exemple précédent – part 4
  - #### 3.2.3.c Ex. regr. mixte (cont. & cat)
    - # Part 4. Test de specification param.
  - On prend un modèle linéaire classique du modèle np qu'on a estimé auparavant

    $$lwage = \beta_0 + \beta_1 Femme + \beta_2 Mari\acute{e} + \beta_3 Educ + \beta_4 Exp + \beta_5 Tenure$$

    Rem. Tenure = durée dans l'emploi actuel
  - Ici, on obtient une p-valeur <5% largement
    - R le modèle linéaire
- Exercice
  - Refaites le test en ajoutant l'exp. quadratique

Sommaire

Qualité de l'ajustement (Goodness-of-Fit)

- Essentiellement, une mesure de R^2 en np

$$R^2 = \frac{\left[\sum_{i=1}^{n} (y_i - \bar{y})(\hat{y}_i - \bar{y})\right]^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2 \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2}$$

  $\hat{y}_i$ est la valeur ajustée de y_i
  - donc $\hat{g}(x_i)$
- $0 \leq R^2 \leq 1$
  - 1 est un ajustement parfait
  - 0 aucun pouvoir prédictif au-delà de la moyenne inconditionnelle
  - Cette mesure est un des produits de la commande npreg
    - accessible par R2 et summary
- Dans le cas d'un modèle linéaire
  - estimé par MCO avec un intercept
  - Cette définition du R^2 produit le même chiffre que la définition classique
    - basée sur les résidus

Régression np – **résumé** de l'approche kernel

- D'abord il faut calculer une bw
  - bw.all<- npregbw
    - on peut préciser ici LL ou LC & la cross-validation
- Ensuite la régression
  - model.np <- npreg(bws=bw.all)
    - summary(model.np) présente la qualité de l'ajustement "R-squared"
  - À ce stade on peut "plot"
    - avec les marges d'erreur
- Puis les tests
  - Significativité des régresseurs
  - Spécification paramétrique
    - Si celle-ci n'est pas rejetée, mieux vaut l'utiliser, car + efficiente

Sommaire

Rappel

- Un panel est une coupe transversale répétée dans le temps t
  - en principe sur les mêmes i
    - sauf attrition, qu'on suppose non-endogène
    - Pour simplifier tous les i sont observés T fois
    - Je ne suis pas sûr que tous les modèles aient été développés pour des panels non-cylindrés
- Lorsque T est grand
  - chaque série de i peut être analysée séparemment en séries temp.
    - qu'on ne voit pas pour np
- Donc, on se place en panels courts : $n \to \infty$ mais T cst
- Les données de panel sont notées $\langle y_{it}, x_{it} \rangle$

Panels non-paramétriques

- Dans le modèle panel linéaire

$$y_{it} = \alpha_i + x_{it}^{'}\beta + \epsilon_{it}$$

  on pouvait accepter que
  - α_i soit la partie constante dans le temps, par i, du terme d'erreur
    - qu'elle soit corrélée avec x_{it} (les "effets fixes")
    - car on pouvait s'en débarrasser par les estimateurs within ou différence 1º
  - Cette hyp. d'additivité de l'hétérogénéité inobservée
    - Fait disparaitre les régresseurs constants dans le temps
- En np, on n'a pas d'hyp. d'additivité
  - Une série d'estimateurs a été proposée

Panel dans np

- Le package np propose l'estimateur suivant
- Soit le modèle np

$$y_{it} = g\left(x_{it}\right) + u_{it}$$

  $g\left(.\right)$ est une fonction lisse inconnue
  x contient q régresseurs
  $E\left(u_{it} | x_{i1}, \ldots, x_{iT}\right) = 0$ exogénéité stricte
- On introduit l'hétérogénéité inobservée constante dans le temps
  - par une variable discrète non-ordonnée

$$\delta_i = i,\ i = 1, ..., n$$

    ce qui introduit de fait n catégories qui ressemblent au α_i du model panel
    - $x_{it} \rightarrow \left(z_{it}, \delta_i\right)$
  - Ces cat nous rapprochent du contexte de la régression np
    - avec un mélange de régresseurs cat. et continus

Panel dans np

- Le paramètre de bandwidth λ pour les catégoriques
  - est limité à [0,1] par contruction dans le package np
- Si $\hat{\lambda}$, la bandwidth estimée est proche de
  - 1 alors $g(z_{it}, \delta_i) = g(z_{it})$
    - δ_i est "smoothed out" : son influence est gommée
    - Les données peuvent être vues comme une grande coupe transversale
  - 0 alors $g(z_{it}, \delta_i) = g_i(z_{it})$
    - Le modèle estime chaque $g_i(.)$ en utilisant seulement la série temporelle de chaque i
  - entre 0 et 1, on est peut-être dans un cas proche aux panel plus standards
    - une partie propre à i,
    - une partie "mélangeable" : les i peuvent être utilisés ensemble pour estimer des pentes communes

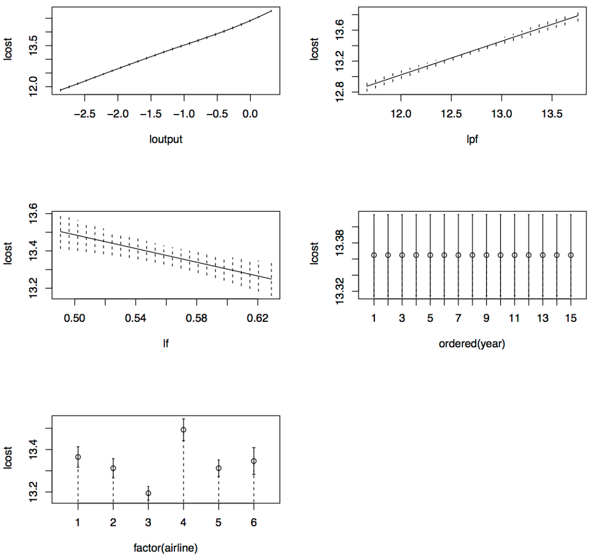
Panel dans np : test

- Si le "régresseur" δ catégorique n'est pas significatif
  - $g(z_{it}, \delta_i) = g(z_{it})$: Les données sont de fait mélangées
    - Pas de diff entre i
  - npsigtest( ) permet ce test (toujours le même)
- Par contre, s'il est significatif
  - Les pentes par i sont différentes $g(z_{it}, \delta_i) = g_i(z_{it})$
  - Donc les données de différents i ne sont pas mélangeables
- L'analyse Panel n'est donc pas complètement implémentée dans np
  - Car l'endogénéité ne parait pas traitée / éliminée comme avec effets fixes
  - mais np apporte un complément d'analyse

Ex. Panel dans np

- #### Ex. regr. panel
- Panel annuel sur les coûts de 6 cies aériennes US
  - 15 ans 1970 à 1984 (cylindré)
  - airline *i* traité comme factor non-ordonné
  - year *t* factor ordonné
  - log of cost lcost, log output (passagers $\times$ miles), log prix fuel lpf, "load factor" lf capacité utilisée moyenne de la flotte
- BW : petite pour "airline"
  - suggère de ne pas mélanger les données
    - Donc de faire pls séries temp
  - npsigtest semble produire des résultats aberrants (significativité extrême)
    - Mais il faudrait un test qui tienne compte de la structure d'hétéroc. & de correl.
- Dans le plot diapo suivante
  - On voit un effet marqué par airline ($\approx \alpha_i$)
  - et des pentes plutôt linéaires
    - au total, plutôt favorable au modèle param. linéaire classique ?

Panel dans np – exemple



Exercice

- Répliquer en utilisant les données Grunfeld du package plm
  - En changeant CV ou kernel (LC / LL)

Sommaire

Régressions np

- On part d'un histogramme lissé par un kernel
  - Estimateur de la fct de densité d'un va continue
  - Importance du bandwidth
    - Dans quel rayon autour de x_0 compte-t-on ?
    - Minimiser la Integrated MSE - par LS Cross-validation
  - Par contre le Kernel optimal est à peu près toujours Gaussien
- Généralisation à
  - v.a. catégorique
  - v.a. multivariée
    - Malédiction de la dimensionalité

Régressions np

- Ces éléments permettent de
  - calculer un estimateur kernel de la densité conditionnelle
  - Une régression est l'espérance de la densité conditionnelle
    - conditionnée sur un ou pls régresseurs
    - Estimateur kernel local constant
    - ou local linéaire
- L'effet marginal $\partial E\left(y|x\right)/\partial x$
  - dépend des niveaux des x et de y

Régressions np

- On peut tester la significativité d'un régresseur
  - Calculer l'équivalent d'un R^2
  - Tester une spécification param. avec les même régresseurs
    - Si on ne la rejette pas, la préférer car + efficiente
- Il existe une version panel
  - Resterait à évoquer l'endogénéité
  - Les séries temp.
  - La théorie existe, mais moins des packages

Devoir # 3 : Régressions

1. 3.1.1 Graphes Effet de la bandwidth dans un histogramme
   - Ajuster le nombre d'intervalles
2. 3.1.1 Graphes Récupérer les données CPS1988 du package AER
   - Essayer de trouver la meilleure représentation en 1D et 2D en changeant la bandwidth et/ou le kernel
3. 3.2.3.a Regr. sim. Gradient MCO du MRL bien spécifié $lm(y\ sin(2 * pi * x))$ et du quadratique $lm(y\ x + I(x^2))$
   - Comparer avec le gradient de la régression np
4. 3.2.3.c Ex. regr. mixte (cont. & cat) Part 2
   - Répliquer avec les données bwages du package Ecdat
5. 3.2.3.c Ex. regr. mixte (cont. & cat) Part 4 Tester une spécification paramétrique
   - Refaites le test en ajoutant l'exp. quadratique

Récapitulatif devoirs np

- Devoir # 1 : tests np
- Devoir # 2 : bootstrap
- Devoir # 3 : régression np

Sommaire

Méthodes semi-paramétriques sp

- Assez populaires
- Combinent des éléments des 2 approches
  - Compromis
  - S'appuient sur des paramètres
    - Donc inconsistant en cas de fausse spécification
- Utiles lorsque
  - La malédiction de la dimensionalité fait que le np ne fonctionne pas bien
  - On veut utiliser un modèle paramétrique pour seulement une partie des régresseurs
  - ...
- Pas une méthodo
  - plutôt un cahier de recettes propres à des cas particuliers

4 cas sp implémentés dans le package np

- Extensions du modèle linéaire
  - Modèles partiellement linéaires
    - Les splines sont un autre cas partiellement linéaire
  - Coefficients aléatoires
- Modèles à index unique
  - *y* dichotomique
  - *y* continu

Sommaire

Modèle partiellement linéaire

- Modèle d'apparence simple
  - Mais le calcul de bandwidth le rend assez lourd
    - Bien plus qu'un modèle paramétrique
    - Du moins quand on base la bandwidth sur les données
- Un modèle sp linéaire

$$y_i = x_i^{'}\beta + g\left(z_i\right) + u_i$$

  x et β ont la signification habituelle avec p régresseurs
  - u n'est pas corrélée avec x ou z
    - mais peut être hétéroscédastique
  - z comprend q régresseurs
  - $g\left(\right)$ n'est pas spécifiée
- L'important est d'obtenir un estimateur consistant de β
  - Puisque quand on l'a, $g\left(\right)$ est estimée par régression np de $y - x\hat{\beta}$ sur $g\left(x\right)$

Estimation de β

- Dans le modèle sp linéaire

$$y_i = x_i^{'}\beta + g(z_i) + u_i$$

on prend l'espérance conditionnelle à z

$$E(y_i|z_i) = E(x_i|z_i)^{'}\beta + g(z_i)$$

en soustrayant les 2 :

$$y_i - E(y_i|z_i) = (x_i - E(x_i|z_i))^{'}\beta + u_i$$

- On ne connait ni $E(y_i|z_i)$ ni $E(x_i|z_i)$
  - Mais on peut les estimer par regr. np de façon consistante
    - Régression car espérance conditionnelle

Estimation de β

- Soit $\hat{E}\left(y_i|z_i\right)$ et $\hat{E}\left(x_i|z_i\right)$ ces estimations
- On écrit

$$\tilde{y}_i = y_i - \hat{E}\left(y_i|z_i\right) \text{ et } \tilde{x}_i = x_i - \hat{E}\left(x_i|z_i\right)$$

  alors

$$\hat{\beta}_{p.linéaire} = \left[\tilde{X}'\tilde{X}\right]^{-1}\tilde{X}'\tilde{Y}$$

- À présent, on réécrit le modèle sp linéaire

$$y_i - x_i'\hat{\beta}_{p.lin} = g\left(z_i\right) + \epsilon_i$$

  où le u_i devient un ϵ_i
  - pour souligner qu'au lieu de β, on utilise $\hat{\beta}_{p.lin}$
  - Ce modèle est alors estimé par la technique np habituelle
    - Mais il est d'un intérêt limité puisque que c'est la partie linéaire qui nous intéresse

Ex. Modèle partiellement linéaire

- Données Wage 1
  - Comme précédemment
- ### Modele partiellement lineaire
  - Commande principale
    - npplregbw pour la bw
    - npplregbw(formula=lwage~factor(female)+ factor(married)+ educ+ tenure|exper, data=wage1)
    - la partie non linéaire va après le | dans la formula
    - npplreg(bw) pour la régression
  - Bug pour le plot ?
- Les t-stats ?
  - npsigtest ne fonctionne pas pour ce cas
  - mais on peut extraire des ec.types pour la partie linéaire
    - et donc calculer des t-stats pour cette partie
    - mais on reste sans savoir si la partie $g(z_i)$ est significative

Exercice Modèle partiellement linéaire

- Au lieu de exper, traiter educ comme non-linéaire
    - Exper en quadratique
- Traiter educ et exper comme non-linéaires

Modèle à coefficients aléatoires

- Coefficients “lissés” ou “variables”

$$\begin{cases} y_i &= \alpha(z_i) + x_i^{'}\beta(z_i) + u_i \\ \text{l@ l@@l@ l@} &= \left(1 + x_i^{'}\right)\begin{pmatrix}\alpha(z_i) \\ \beta(z_i)\end{pmatrix} + u_i \\ &= w_i^{'}\gamma(z_i) + u_i \end{cases}$$

- en mots : les coef α et β sont des fonctions de z
  - non-spécifiées & lisses
    - comme ça, on va pouvoir les estimer par np
- Pourquoi choisir certaines variables comme z ?
  - P.e. des pentes différentes selon certains niveaux de factors
  - Les rendements de l’éduc, de l’exp dépendent-ils de si on est marrié ? p.e.

Modèle à coefficients aléatoires

- Prémultiplier par w_i
  - et prendre l'espérance conditionelle à z_i

$$E\left[w_i y_i | z_i\right] = E\left[w_i w_i^{'} | z_i\right] \gamma\left(z_i\right) + E\left[w_i u_i | z_i\right]$$

$\gamma\left(z_i\right)$ sort de l'espérance car il est déjà conditionnel à z_i

- On se retrouve dans une situation proche du modèle partiellement linéaire
  - On peut exprimer $\gamma\left(z_i\right)$ comme dans MCO

$$\gamma\left(z_i\right) = \left(E\left[w_i w_i^{'} | z_i\right]\right)^{-1} E\left[w_i y_i | z_i\right]$$

  - Les espérances conditionnelles sur z_i peuvent être estimées par np
    - En utilisant l'ensemble de l'échantillon, pas L à L

Ex. Modèle à coefficients aléatoires

- Données Wage 1
- ### Modele a coefficients aleatoires
  - Commande principale
    - npscoefbw pour la bw
    - npscoefbw(formula=lwage~ educ+ tenure+ exper+ expersq|factor(female)+factor(married))
    - Les z sont spécifiés après la |
    - npscoef(bw,betas=TRUE) pour les coef
  - Assez peu d'output
  - Comment interpréter les moyennes des coefs aléatoires ?
- Exercice
  - Essayer de calculer les t-stats (sans garantie)

Sommaire

Index unique / "Single index"

- Modèles de forme

$$y_i = g\left(X_i'\beta\right) + u_i$$

  où les notations sont celles du MRL
  avec q régresseurs
  et $E\left(u_i|X_i\right) = 0$
- $X_i'\beta$ est une forme linéaire (= un index)
  - dont on ne connait pas les paramètres β
- Lorsque $g\left(\right)$ est
  - identité : MRL
  - La fonction de probabilité Logistique ou normale : Logit / probit

Single index semi-paramétriques

- On a un modèle semi-paramétrique lorsque
  - $g(.)$ est une fonction inconnue, non-spécifiée
  - "Semi" car paramètres β
    - inconnus également
- + flexible que paramétrique
  - moins susceptible à la malédiction de la dimensionalité
    - car $x_i' \beta$ est de fait un seul régresseur
    - On appelle cela une "réduction de la dimensionalité"
    - Donc convergence + rapide de l'estimation

Identification

- Avant de pouvoir estimer,
  - il faut que le modèle soit identifiable
  - c'est-à-dire qu'il ait la capacité de produire une estimation
    - et que ce soit de manière unique
- Plus précisément, si on a ∞ obs.
  - peut-on théoriquement découvrir les vrais paramètres (ou fonction) du modèle de manière unique ?
- Ex. 1. le MRL $y = \alpha + X\beta + \epsilon$, $E\left(\epsilon|x\right) = 0$ est identifiable
  - ssi $X'X$ est invertible
- Ex. 2. Modèle probit $\Pr\left(\alpha + X\beta + \epsilon < 0\right)$
  - Soit $y = 1$ ssi $\alpha + X\beta + \epsilon < 0$, $\epsilon \sim n\left(0, \sigma^2\right)$
  - Comme $\left(\alpha + X\beta + \epsilon < 0\right) \iff a\left(\alpha + X\beta + \epsilon < 0\right)$, $a > 0$,
  - il résulte que σ^2 n'est pas identifiable
  - Probit est partiellement identifiable avec ce type de données

Identification des modèles single-index

- Le MRL comme Probit sont 2 cas particuliers de single index
  - On voit donc 2 conditions d'identification
  - Il faut aussi que $g()$ ne soit pas une fonction constante
    - $g(X\beta) = c \, \forall X$
- Soit 2 constantes γ quelconque et $\delta \neq 0$, alors
  - $E(Y|X = x) = g\left(x'\beta\right)$ et
  - $E(Y|X = x) = g^*\left(\gamma + \delta x'\beta\right)$
    - sont dites "observationnellement équivalentes"
  - Même si on connaissait la distribution de (Y, X)
    - On ne pourrait pas distinguer ces 2 modèles (savoir lequel est le bon)
  - Il faut donc des restrictions additionnelles
    - pour identifier β et $g()$

Identification des modèles single-index

- La restriction sur γ est appelée **normalisation de localisation**
  - Le point glisse sur le support
  - Essentiellement la moyenne
  - Est obtenue en imposant que X ne contienne pas de constante
    - (y compris des combinaisons lin. des col. de X)
- Sur δ : **normalisation d'échelle**
  - Essentiellement la variance, la dispersion
  - Est obtenue en imposant qu'un des coef. de β soit égal à un
- Il faut aussi que
  - $g()$ soit différentiable (sans preuve)
  - un au moins des éléments de X soit une variable continue
    - diapo suivante

Single-index à variables catégoriques uniquement

- Soit $X = (X_1, X_2)$ à supports $\{0, 1\}$ pour les 2 éléments de X
  - On normalise le coef de X_1 à 1
    - Normalisation d'échelle
  - Le modèle $E(Y|X = x) = g\left(x'\beta\right)$ devient
    - $E(Y|X = x) = g(x_1 + x_2\beta_2)$
- Supposons que les données soient t.q.

(x_1, x_2)	$E(Y\|X = x)$	$g(x_1 + x_2\beta_2)$
$(0, 0)$	0	g(0)
$(1, 0)$	0.1	g(1)
$(0, 1)$	0.3	g(β_2)
$(1, 1)$	0.4	g($1 + \beta_2$)

- Comme on peut choisir à la fois $g()$ et β_2
  - il y a ∞ de solutions : pas identifié

2 cas sont traités dans le package np

- Ichimura : y est continu
- Klein & Spady : y est une dichotomique
  - Intéressant car on n'a pas d'équivalent logit/probit en np

Ichimura : y est continu

- Si on connaissait $g()$
  - On pourrait estimer β par Moindres Carrés Non-Linéaires

$$\hat{\beta} \equiv \arg\min_{\beta} \frac{1}{n} \sum_{i} W_i \left(y_i - g\left(X_i' \beta \right) \right)^2$$

    W_i est un poids, sur lequel on ne s'attarde pas ici
  - À cause de la non-linéarité de $g()$:
    - Il n'y a pas de solution analytique
    - On n'a que des approches numériques
- Mais on ne connait pas $g()$
  - Et on ne peut pas appliquer un estimateur Kernel
    - car on ne connait pas β
- Ichimura propose une modification de MC non-lin.
  - dans laquelle $g()$ est estimé par un kernel modifié

Ex. Ichimura

- Même exemple avec wage1 qu'on a déjà utilisé
- ### 3.3.a Ichimura
- L'estimateur d'Ichimura est invoqué en 2 étapes comme toujours
  - npindexbw pour la bandwidth
  - npindex pour les coefficients
- La fonction $g()$ n'est pas dans l'output
  - puisqu'on l'estime de façon np
  - Les coefficient sont comparables avec un modèle linéaire
    - au sens où on peut comparer leur impact relatif
  - Les gradients peuvent être "plot" par la commande usuelle
    - Mais l'interprétation est difficile, voir le script
- La méthode est intensive en ordinateur : long
- Exercice. La commande semble accepter une matrice vcov des coefficients, trouver comment estimer des t-stat

Estimateur de Klein & Spady

- Ici, y est dichotomique
  - Alors $g\left(x_i'\beta\right) = \Pr\{Y = 1|X = x_i\}$
- Si $g()$ était connue,
  - alors l'estimation se ferait par max. vraisemblance
  - Avant de poursuivre, on va faire un rappel sur MV

Maximum de vraisemblance

- La fonction de densité de probabilité d'une va y
  - conditionnellement à des paramètres θ
  - est notée $f(y|\theta)$
- Si on a un éch. de n obs. iid de cette va
  - Alors, on peut définir la densité jointe de l'éch. comme le produit des densités de chaque obs.

$$f(y_1, ..., y_n|\theta) = \prod_{i=1}^{n} f(y_i|\theta) = L(\theta|y)$$

  - Cette densité jointe est appelée **fonction de vraisemblance** (**likelihood**)
  - Très souvent, on prend le log

$$\ln L(\theta|y) = \sum_{i=1}^{n} \ln f(y_i|\theta)$$

Maximum de vraisemblance

- Il est souvent nécessaire de généraliser pour introduire des régresseurs
- Soit le MRL $y_i = x_i\beta + \epsilon_i$
  - On suppose que ϵ est normale
    - avec moyenne 0 & variance σ^2
  - Donc $y_i|x_i$ est normale
    - avec moyenne $\mu_i = x_i\beta$ et variance σ^2
- Donc, les obs. de la va **ne sont pas iid**
  - Elles ont des moyennes différentes
  - Mais elles restent indépendantes
  - Et on peut les standardiser pour qu'elles aient la même moyenne
  - Ça donne la fonction de vraisemblance

$$\ln L\left(\theta|y,X\right) = \sum \ln f\left(y_i|x_i,\theta\right) = -\frac{1}{2}\sum_{i=1}^{n}\left[\ln\sigma^2 + \ln\left(2\pi\right) + \left(y_i - x_i\beta\right)^2/\sigma^2\right]$$

Maximum de vraisemblance : Probit

- Le même MRL avec un seul régresseur + 1 cst

$$y_i = \beta_1 + \beta_2 x_i + \epsilon_i$$

  où $\epsilon_i | x_i \sim n\left(0, \sigma^2\right)$
- Contexte d'un achat important, p.e. une voiture
  - x_i est le revenu de i
  - y_i est la $\neq$ entre la disposition à payer p_i^*, et le prix de la voiture p_i
- On n'observe pas y_i mais seulement si i achète la voiture ou non
  - Soit $y_i^* = 1$ lorsque $y_i = p_i^* - p_i > 0$
  - Et donc $y_i^* = 0$ sinon
    - (quels sont ces i qu'on "n'observe pas" acheter ?)

Maximum de vraisemblance : Probit

- La proba d'achat est

$$
\begin{aligned}
\Pr\left\{y_i^* = 1|\beta_1, \beta_2, \sigma, x_i\right\} &= \Pr\left\{y_i > 0|\beta_1, \beta_2, \sigma, x_i\right\} \\
&= \Pr\left\{\beta_1 + \beta_2 x_i + \epsilon_i > 0|\beta_1, \beta_2, \sigma, x_i\right\} \\
&= \Pr\left\{\epsilon_i > -\beta_1 - \beta_2 x_i|\beta_1, \beta_2, \sigma, x_i\right\} \\
&= \Pr\left\{\epsilon_i/\sigma > \left(-\beta_1 - \beta_2 x_i\right)/\sigma|\beta_1, \beta_2, \sigma, x_i\right\} \\
&= \Pr\left\{z_i > \left(-\beta_1 - \beta_2 x_i\right)/\sigma|\beta_1, \beta_2, \sigma, x_i\right\}
\end{aligned}
$$

où z_i a une dist. normale standard

- La proba de ne pas acheter est un moins cette proba

Probit : identification par normalisation

- Donc la fonction de vraisemblance est

$$\prod_{i=achat} \left[\Pr\left\{ach|\beta_1,\beta_2,\sigma,x_i\right\}\right] \times \prod_{i=pas\ ach} \left[1-\Pr\left\{ach|\beta_1,\beta_2,\sigma,x_i\right\}\right]$$

- Souvent réécrite comme

$$\prod_i \left[\Pr\left\{ach|\beta_1,\beta_2,\sigma,x_i\right\}^{y_i}\right]\left[1-\Pr\left\{ach|\beta_1,\beta_2,\sigma,x_i\right\}\right]^{(1-y_i)}$$

- Les paramètres β_1,β_2,σ de ce modèle **ne sont pas identifiés** :
  - S'ils sont multipliés par une même constante non-nulle
    - le nouveau modèle est observationnellement équivalent
    - $\Pr\{ach\}$ et la fct de vrais. ne changent pas
- La **normalisation** généralement utilisée est $\sigma = 1$

Interprétation

- Avec une va discrète, $f(y_i|\theta)$ est la proba d'observer y_i conditionnellement à θ
  - La fct de vraisemblance est la proba d'observer l'éch.
- On suppose que l'éch. qu'on a est le + probable
  - Sorte de principe de médiocrité
- Si on choisit θ de sorte à maximiser la fonction de vraisemblance
  - alors on max la proba de l'éch.
  - Ce qui correspond bien à l'hyp de médiocrité

Maximum de vraisemblance : notes

- Cette maximisation produit l'estimateur max. de vraisemblance
  - Généralement numériquement + simple de max. la log-vraisemblance
  - Comme ln est monotone, c'est le même max
- La condition de 1º ordre de la max

$$\partial \ln L\left(\theta | data\right) / \partial \theta = 0$$

est appelée équation de vraisemblance

Maximum de vraisemblance : propriétés

- Si l'hypothèse de distribution est correcte
  - dans le probit : $\epsilon_i | x_i \sim n\left(0, \sigma^2\right)$
  - Alors MV a de très bonnes propriétés
  - **Notation** : $\hat{\theta}$ estimateur ; θ_0 vraie valeur ; θ toute autre valeur
- **Consistance** : $plim\, \hat{\theta} = \theta_0$
- **Normalité asymptotique** : $\hat{\theta} \sim N\left[\theta_0, \left\{I\left(\theta_0\right)\right\}^{-1}\right]$
  - $I\left(\theta_0\right) = -E\left[\partial^2 \ln L / \partial\theta_0 \partial\theta_0^{'}\right]$ est dite **information matrix**
- **Asymptotiquement efficient**
  - $\hat{\theta}$ atteint la borne de Cramer–Rao pour les estimateurs consistants
- **Invariance** : l'estimateur de $\gamma_0 = c\left(\theta_0\right)$ est $c\left(\hat{\theta}\right)$
  - si $c\left(\theta\right)$ est continue et différentiable
- MV n'a que des propriétés asymptotiques
  - En petits éch. il peut être biaisé/inefficient

Retour à Klein & Spady

- Un estimateur sp pour le cas probit
  - $g\left(x_i'\beta\right) = \Pr\{Y = 1|X = x_i\}$
- Si $g()$ était connue,
  - alors l'estimation se ferait par max. vraisemblance
    - p.e. probit, on suppose que $g()$ est la probabilité normale, donc

$$\ln L(\beta|y,x) = \sum_i [y_i \ln \Pr\{y_i = 1|\beta, x_i\}] [(1 - y_i) \ln (1 - \Pr\{y_i = 1|\beta, x_i\})]$$

- $g()$ n'étant pas connue
  - on a le même problème qu'avec Ichimura
  - Klein & Spady proposent un estimateur Kernel modifié

Ex. Klein & Spady

- Données sur le poids à la naissance (détails dans le script)
- ### 3.3.b Klein & Spady
  - Voir le script, c'est long mais simple
  - Matrice de confusion

			Prédit 0	Prédit 1
Réel	0	Logit	119	11
		K&S	119	11
			vrais ⊖	faux ⊕
	1	Logit	34	25
		K&S	22	37
			faux ⊖	vrais ⊕

- Léger avantage à K&S

Exercice

- Est-il possible de calculer des t-stats ?

Sommaire

Intro

- Local Polynomial Regression Fitting / locally weighted smoothing
  - Aussi appelé LOWESS (Locally Weighted Scatterplot Smoothing)
    - Dans R, lowess ne prend qu'un seul régresseur
- Approche "local polynomial kernel" qu'on a déjà vue
- Scatterplot smooting est un autre nom
  - attention aux régresseurs manquants avec cette interprétation
- loess est une alternative au package np
  - Mais fait moins de choses

commande

loess(formula, data, span = 0.75, degree = 2, parametric = FALSE, family = c("gaussian", "symmetric"), ...)

- formula permet de spécifier le modèle
  - Ne prend pas de factor !
  - Pas plus de 4 régresseurs !
- span : la proportion de points de l'éch. utilisés pour le "local fitting" (ad hoc)
- degree =1 indique locally linear regressions
  - =2 (défaut) locally quadratic regressions
- parametric : permet de spécifier certains régresseurs comme paramétrique
  - Ça fait juste span=1 pour ce ou ces régresseurs-là, mais ne produit pas un coefficient
  - Ça n'est pas comme les modèles partiellement linéaires
- family est le kernel

Représentation graphique

- Ne présente pas de paramètre, comme toujours
  - Les résultats de la régression sont visibles graphiquement
    - Mais il faut détailler le plot
    - `fit.wage1 <- matrix(predict(model.loess, newwage1), 25, 25)` prédiction du modele
    - `persp(edu, exp, fit.wage1, theta=0, phi=30, ticktype="detailed", xlab="Education", ylab="Experience", zlab="Wage", expand=2/3, shade=0.5)` le plot en 3D
    - Changer `theta` pour faire tourner l'image à l'horizontale
    - `phi` pour la verticale
  - On voit bien que le rendement de l'expérience est croissant avec l'éducation
    - La forme est assez intéressante

Significativité

- Pas de commande directe, il faut
  - répliquer la régression moins un régresseur
  - calculer un F-test sur base des "residual sums of squares" dans les 2 régressions
  - anova(modèle moins un régresseur, modèle initial)
    - Il reste une hypothèse de normalité des résidus derrière le test en F
  - Les régresseurs sont tous très significatifs
- Il ne semble pas y avoir de "goodness-of-fit" associé à loess
- Exercice. Répliquer l'analyse avec les données Airq du package Ecdat

Smoothing d'une série temporelle

- Une application graphiquement intéressante
- Data economics de ggplot2
  - uempmed median duration of unemployment, in weeks,
  - date Month of data collection
    - Remplacé par index (=row nbr) pour loess qui n'accepte que du numérique
  - Estimation à plusieurs niveaux de span
  - Plot des prédictions à plusieurs niveaux de span