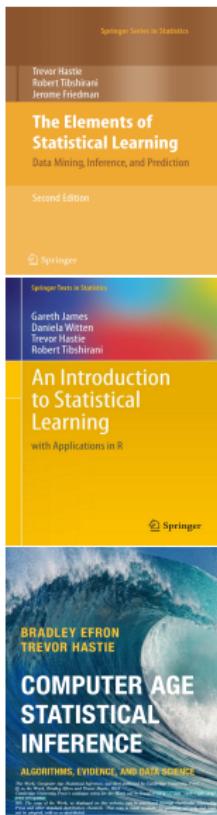


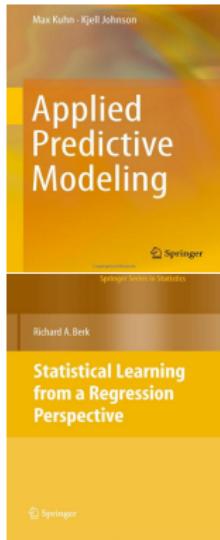
Econométrie des Big Data

(2020-2021)

Olivier DARNÉ



- **Hastie, Tibshirani et Friedman**, *The Elements of Statistical Learning*, Springer. [\[pdf\]](#)
- **James, Witten, Hastie et Tibshirani**, *An Introduction to Statistical Learning*, Springer. [\[pdf\]](#)
- **Efron et Hastie**, *Computer Age Statistical Inference*, Cambridge. [\[pdf\]](#)



- [Kuhn et Johnson](#), *Applied Predictive Modelling*, Springer.
[\[website\]](#)
- [Berk](#), *Statistical Learning from a Regression Perspective*, Springer.

Économétrie, Machine Learning et Big Data

L'économétrie et le Machine Learning se ressemblent par certains points car ils se servent tous les deux de bases de données avec comme objectif de prédire.

Pourtant, ils sont bien différents dans leur interprétation, l'économétrie reposant davantage sur des théories économiques et financières et des modèles paramétriques contrairement au Machine Learning.

Charpentier, Flachaire et Ly (2018). [Économétrie et Machine Learning. Économie et Statistique](#), No 505-506. [\[pdf\]](#)

Définition des big data

Une différence doit être faite entre les grandes "données officielles" et les "données alternatives":

- **Les données officielles**

- publiées par les instituts nationaux de statistique ou les banques centrales
- données désagrégées ou granulaires (individuelles) (micro et macro)

- **Les données alternatives → big data**

- les réseaux sociaux (Facebook, Twitter ...)
- les recherches sur internet (Google avec *Google Trends*, Bing ...)
- les données du Web (*Web scraped data*),
- les données de caisse (*scanner scraped data*),
- le *crowdsourcing* avec les téléphones mobiles
- les données des cartes de crédit
- les données du e-commerce
- ...

Les 5 V

Le passage des données officielles aux **big data** ⇔ **des 3 V aux 5 V**

- (1) **Volume** : les énormes quantités de données générées (maintenant par seconde et leur contenu informationnel)
- (2) **Vitesse** ou **Vélocité** (*Velocity*) : la vitesse de leur création, collecte, transmission et analyse
- (3) **Variété** (*Variety*) : les différences de natures, sources, formats et structures
- (4) **Véracité** (*Veracity*) : leur fiabilité, validité, qualité, exactitude, précision
- (5) **Valeur** (*Value*) : la capacité à générer des profits et leur coût

Type de big data

Doornik et Hendry (2015) proposent de distinguer 3 principaux types de big data

- *Tall*: pas de trop nombreuses variables (p) mais beaucoup d'observations (N) : $N \gg p$
- *Fat*: de nombreuses variables mais peu d'observations : $p > N$
- *Huge*: de nombreuses variables et observations : $N > p$

Cette classification permet d'associer les méthodes économétriques et ML appropriées à chaque type de données.

Dans ce cours on s'intéressera aux bases de données *Tall* et *Huge*

Avantages et inconvénients des big data

● Avantages

- disponibles rapidement
- peu de révisions
- information granulaire en dimensions temporelle (N) et transversale (p)

● Inconvénients

- la disponibilité des données : la continuité de la diffusion des données n'est pas garantie
- la date de disponibilité des données : souvent très récente : échantillon trop court
- la fracture numérique (*digital divide*) : problème de représentativité (biais de sélection)
- la taille et la qualité des données d'internet dans le temps
- fiabilité des données
- problèmes des **irrégularités** des données : outliers, ruptures structurelles, valeurs manquantes, saisonnalité, filtres

Problèmes méthodologiques

L'essor des big data doit s'accompagner d'une attention particulière aux méthodologies utilisées et peuvent engendrer des erreurs statistiques

Pour Doornik et Hendry (2015) : “*an excess of false positives, mistaking correlations for causes, ignoring sampling biases and selecting by inappropriate methods*”

Une autre critique est le *big data hubris* (hubris : excès, démesure, orgueil) formulée par Lazer et alii (2014) associée à l'hypothèse implicite que, “*big data are a substitute for, rather than a supplement to, traditional data collection and analysis*”

Analyse exploratoire des données

Avant toute chose les données doivent subir une analyse exploratoire

- Statistiques descriptives
- Graphiques des données
- Corrélations
- Distributions des données
- Outliers
- Stationnarité
- Classification (ACP, K-means ...)

Caractéristiques des modèles pour le Big Data

Le modèle de régression linéaire sous forme matricielle est donné par

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

- \mathbf{Y} : vecteur ($N \times 1$) de la variable expliquée (stationnaire)
- \mathbf{X} : vecteur ($N \times p$) de variables explicatives (stationnaires)
- $\boldsymbol{\beta}$: vecteur ($p \times 1$) de coefficients
- $\boldsymbol{\varepsilon}$: vecteur ($N \times 1$) des résidus

Si p est grand ($p >> N$) \Rightarrow Big Data \Rightarrow modèles à grande dimension \Rightarrow Inférence statistique non optimale et inefficace

- Trop de paramètres à estimer
- Haut degré d'incertitude dans les estimations
- Sur-ajustement et mauvaise précision hors échantillon

Comment gérer cette dimension p ?

Approche “Économétrique”

Méthode GETS

Sélection des variables

Il existe différentes méthodes séquentielles de sélection :

- Méthode de recherche exhaustive (*exhaustive search*) : nombre de sous-modèles possibles pour N variables explicatives : $2^N \Rightarrow$ coût algorithmique prohibitif pour N grand, même modéré
- Méthode de recherche ascendante (*forward*) : on part du modèle constant (aucune variable explicative), et on ajoute de manière séquentielle les variables qui permettent d'optimiser le critère (maximiser le \bar{R}^2 , minimiser les autres critères comme les critères d'information AIC ou BIC), ou d'accepter la validité du modèle (test)
- Méthode de recherche descendante (*backward*) : même principe, mais on part du modèle complet et on élimine une à une les variables.
- Méthode de recherche progressive ou pas-à-pas (*stepwise*) : algorithme mixte qui ajoute ou supprime une variable du modèle à chaque étape

Critiques

- coût **algorithmique prohibitif** pour N grand, même modéré
- **risque de dépendance au sentier** (*path dependency*) :
 - le choix de la variable à l'étape t est conditionné par toutes les étapes précédentes (de 1 à $t - 1$)
 - selon le chemin pris, les modèles finaux ne sont donc pas les mêmes

Sélection et donc estimation finalement peu satisfaisantes, instables, surtout inadaptées au cas $p \geq N$

Deux principales stratégies ont été proposées pour la **sélection automatique de modèles** :

(1) L'*approche specific-to-general* (SETP) basée sur une régression *stepwise* ou *forward selection* avec les logiciels

- RETINA (Perez-Amaral, Gallo et White, 2003) (*relevant transformation of the inputs network approach*)
- QuickNet (White, 2006) (réseaux de neurones)

(2) L'*approche general-to-specific* (GETS), proposée par David Hendry à la *London School of Economics* (LSE), est une méthode de la famille des *backward selections* avec les logiciels

- PcGets (Hendry et Krolzig, 1999 ; Krolzig et Hendry, 2001)
- Autometrics (Doornik and Hendry, 2007 ; Doornik, 2009)
- AutoSearch (Sucarrat, 2015) et gets (Sucarrat et alii, 2018)

Algorithmes basés sur l'approche GETS

Hoover et Perez ont été les premiers à proposer une approche algorithmique de la méthode GETS en introduisant une recherche à **sentiers multiples** (*multi-path*)

- partir d'un modèle très général
- procéder, selon plusieurs modèles possibles, à l'élimination successive de variables non significatives, à condition qu'un certain nombre de tests de spécification soient acceptés à chaque étape
- obtenir un modèle qui soit à la fois bien spécifié, cad qu'une variable explicative importante du phénomène à étudier n'a pas été omise, et parcimonieux, cad qu'il ne contienne pas de variables inutiles

Hendry et Krolzig (1999) et Krolzig et Hendry (2001) ont proposé une **2nd génération de l'algorithme** avec le logiciel **PcGets**, en introduisant plusieurs améliorations (sélection du modèle final par un critère BIC, un seuil plus bas pour la significativité des variables, ...)

Doornik and Hendry (2007) et Doornik (2009) ont proposé une **3ème génération de l'algorithme** avec **Autometrics** (sous Ox), notamment en gérant le cas $p > N$ avec le l'algorithme *cross-block* (Hendry et Krolzig, 2004)

Sucarrat et Genaro (2009) généralisent la méthode de Hoover et Perez avec le logiciel **AutoSearch** (Sucarrat, 2015) et la dernière version **gets** (Sucarrat et alii, 2018) (sous R), notamment en rendant possible l'existence d'un autorégressif (variable endogène retardée) en plus des variables explicatives

Le package **gets** procède de la manière suivante :

Step 1. Formuler un modèle non restreint général (GUM, *General Unrestricted Model*) passant une série de tests de diagnostiques :

- Tests d'autocorrélation sur les résidus standardisés (Ljung et Box, 1979)
- Tests d'hétéroscléasticité conditionnelle sur les résidus standardisés au carré (Ljung et Box, 1979)
- Test de normalité (Jarque et Bera, 1980)

Step 2. Eliminer de manière successive (*backwards elimination*) les variables non significatives du GUM à partir d'un seuil de significativité $\alpha = 5\%$

Step 3. La validité de chaque suppression est vérifiée par rapport à :

- une série de tests de diagnostiques (cf. Step 1)
- un test à hypothèses multiples par rapport au GUM (PET, *Parsimonious Encompassing Tests*)

Step 4. Sélectionner, parmi les modèles terminaux, la spécification la mieux ajustée selon un critère d'ajustement (BIC).

Méthode GETS et package R

Le package **gets** implémente la méthode GETS

- Si $p > N \Rightarrow$ le package (actuel) ne fonctionne pas car il est fondé sur des régressions linéaires
- Pratique : il peut arriver que les tests de diagnostiques (autocorrélation et hétéroscédasticité conditionnelle) soient trop restrictifs et soient ne sélectionnent aucune variable ou bien un très petit nombre.
 - gets avec tests
 - gets sans tests (test ARCH)
- Contrainte : $p < 89$

Sélection de variables et packages R

Le package **leaps** avec la fonction `regsubsets` applique l'approche de la sélection *Best subset* (meilleur sous-ensemble) proposée par Miller (1990)

- Cette approche consiste à tester toutes les combinaisons possibles de variables explicatives et de sélectionner le meilleur modèle selon des critères statistiques (R^2 , \bar{R}^2 , BIC et C_p e Mallows)
- La sélection du modèle peut se faire aussi par *exhaustive search, forward, backward, stepwise ou sequential replacement*

Le package **glmulti** avec la fonction `glmulti` propose une approche de sélection automatique de modèles basée sur un algorithme génétique proposée par Calcagno et de Mazancourt (2010). [\[pdf\]](#)

Voir *Algorithmes de sélection de variables dans une régression de Malouche* [\[website\]](#)

Figure: Séries en niveau

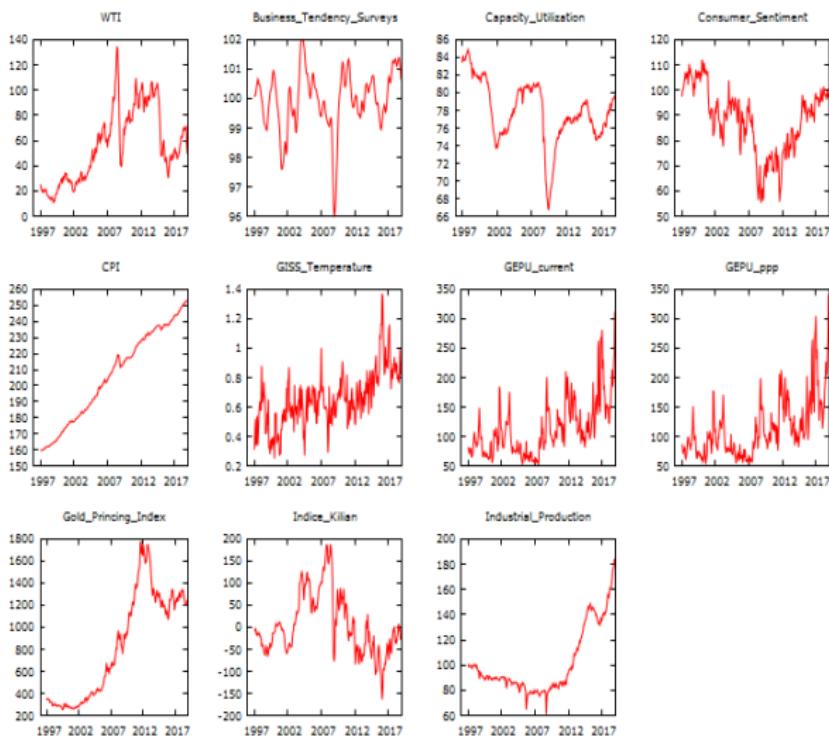


Figure: Séries en niveau

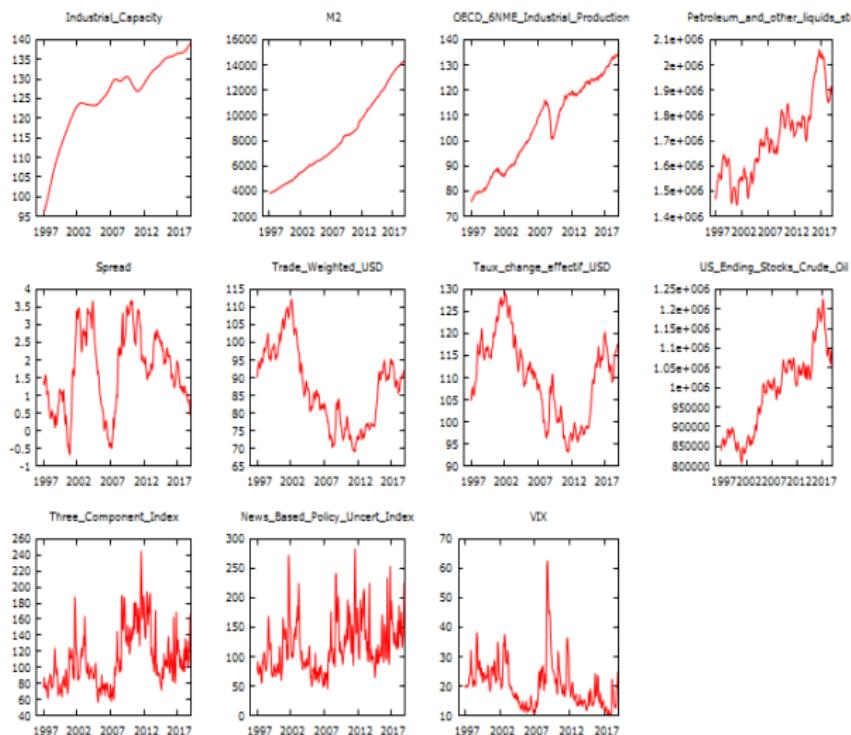


Figure: Séries en différence première

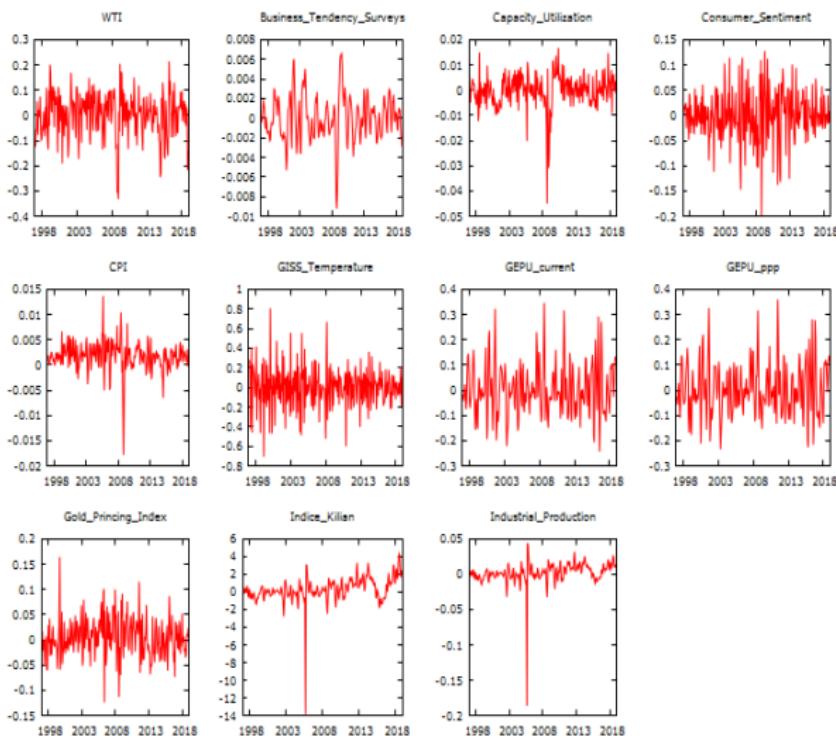


Figure: Séries en différence première

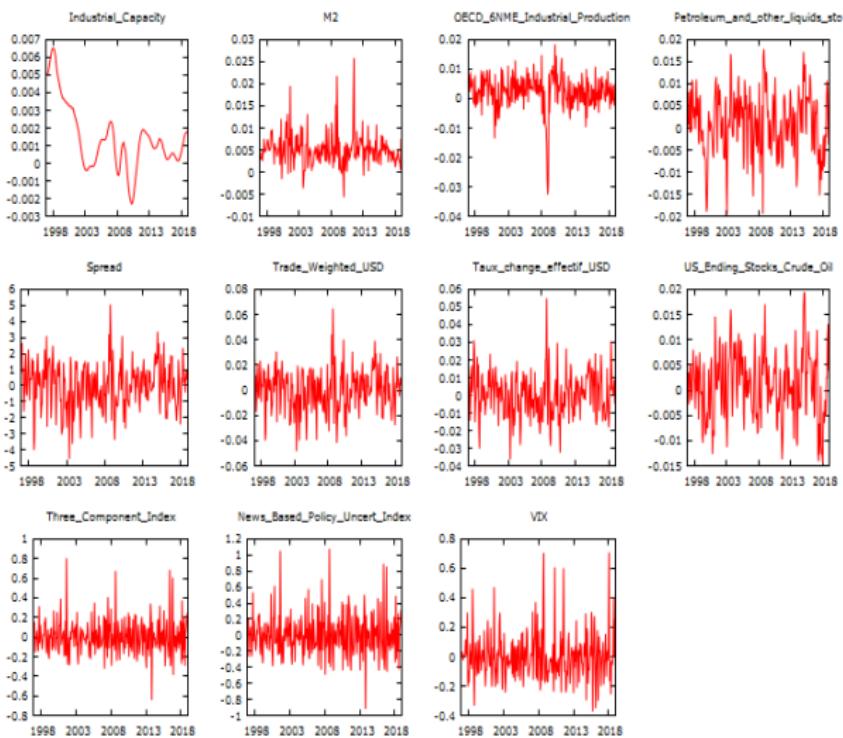


Figure: Matrice des corrélations (linéaires) entre les variables

	WTI	Business_Tendancy_Surveys	Capacity_Utilization	Consumer_Sentiment	CPI	GISS_Temperature	GEPU_current	GEPU_ppp	Gold_Pricing_Index	Kilian_Index	Industrial_Production	M2	OECD_GNME_Petroleum	Spread	Trade_Weighted_US_Ending	Taux_change_effectif_USD	Three_Component_Index	News_Based_Policy_Uncert_Index	VIX			
WTI	1.000	0.397	0.117	-0.023	0.557	-0.070	-0.122	-0.116	0.125	-0.058	-0.046	-0.097	-0.216	0.274	-0.317	-0.332	-0.349	-0.287	-0.268	-0.054	-0.048	-0.195
Business_Tendancy_Surveys	1.000	0.187	0.159	0.264	-0.024	-0.298	-0.272	0.108	0.023	0.017	-0.127	-0.235	0.314	0.015	-0.233	-0.253	-0.255	-0.004	-0.148	-0.150	-0.269	
Capacity_Utilization		1.000	-0.114	0.070	-0.056	0.000	0.015	0.050	0.284	0.281	-0.174	-0.235	0.508	-0.098	-0.014	-0.123	-0.030	-0.136	-0.086	-0.070	0.118	
Consumer_Sentiment			1.000	-0.165	-0.023	-0.209	-0.188	-0.109	0.183	0.202	0.018	-0.233	0.020	0.058	0.057	0.058	-0.043	0.103	-0.117	-0.145	-0.252	
CPI				1.000	0.029	0.032	0.028	0.239	-0.213	-0.244	-0.026	-0.108	0.260	-0.366	-0.278	-0.302	-0.158	-0.254	0.082	0.099	-0.026	
GISS_Temperature					1.000	-0.022	-0.017	0.024	-0.021	-0.029	0.025	0.023	-0.052	0.087	-0.042	-0.052	-0.020	0.057	0.029	0.034	0.017	
GEPU_current						1.000	0.985	0.053	-0.070	-0.108	0.013	0.137	-0.017	-0.105	0.131	0.145	0.284	-0.052	0.628	0.648	0.470	
GEPU_ppp							1.000	0.047	-0.046	-0.085	0.008	0.137	-0.009	-0.094	0.137	0.150	0.285	-0.065	0.587	0.605	0.452	
Gold_Pricing_Index								1.000	-0.031	-0.002	-0.172	0.102	0.087	-0.016	-0.451	-0.462	-0.413	-0.040	0.033	0.046	0.073	
Indice_Kilian									1.000	0.962	-0.069	-0.061	-0.010	0.084	0.024	0.023	-0.020	-0.014	-0.079	-0.108	0.006	
Industrial_Production										1.000	-0.059	-0.033	-0.007	0.122	-0.002	-0.008	-0.051	0.022	-0.112	-0.142	-0.012	
Industrial_Capacity											1.000	0.158	-0.052	0.049	0.161	0.152	0.192	-0.035	0.008	0.010	0.039	
M2												1.000	-0.374	0.081	0.033	0.031	0.064	0.065	0.150	0.140	0.225	
OECD_GNME_Industrial_Production													1.000	-0.190	-0.115	-0.120	-0.152	-0.176	0.028	0.027	-0.018	
Petroleum_and_other_liquids_stocks_US														1.000	0.118	0.108	0.059	0.698	-0.108	-0.116	-0.062	
Spread															1.000	0.993	0.888	0.109	0.042	0.038	0.098	
Trade_Weighted_USD																1.000	0.895	0.104	0.040	0.036	0.109	
Taux_change_effectif_USD																	1.000	0.066	0.165	0.175	0.245	
US_Ending_Stocks_Crude_Oil																		1.000	0.000	-0.019	-0.060	
Three_Component_Index																		1.000	0.969	0.360		
News_Based_Policy_Uncert_Index																			1.000	0.370		
VIX																				1.000		

```
library(tidyverse)

#BDD
dlbase <- read_excel("c:\\R\\data\\dlbase.xlsx", sheet = "dlbase")
# dlbase <- read_excel(file.choose(), col_names=TRUE)
dlbase <- data.frame(dlbase)
summary(dlbase)
str(dlbase)
training_dlbase <- dlbase
data.frame(training_dlbase)

# standardized y and x (centered and standardized)
y <- data.frame(training_dlbase) %>%
  select(ipiman) %>%
  scale(center = T, scale = F) %>%
  as.matrix()
x <- data.frame(training_dlbase) %>%
  select(-ipiman) %>% # on retire la variable à expliquer y
  as.matrix()
```

```
# GETS
library(lgarch)                                # Gets modelling
library(gets)

# convert tibble in matrix for the function arx
class(dlbase[,2:11])                           # tibble
mX = data.matrix(training_dlbase[,2:11])         # warning: l'argument 'size' ne doit pas dépasser nrow(x) = 89

# ARX model with AR(1)
model <- arx(training_dlbase$pib, mc = T, ar = 1, mxreg = mX[, 1:10], vcov.type = "ordinary")
model

# GETS modelling
getsm <- getsm(model)
getsm

# GETS betas
coef.arx(getsm)

# Get the name of relevant variables
names_mX <- names(coef.arx(getsm))
names_mX <- names_mX[-1]                         # on retire le ar1
names_mX
```

```
# GETS modelling without ARCH test
getsm2 <- getsm(model, arch.LjungB=NULL)
getsm2

# GETS without AR(1)
mwithoutar <- arx(training_dbase$pib, mc = T, ar = NULL, mxreg = mX[, 1:10], vcov.type = "ordinary")
mwithoutar

# isat function
yy <- dbase[,1]
isat(yy, sis=TRUE, iis=FALSE, plot=TRUE, t.pval=0.005)
isat(yy, sis=FALSE, iis=TRUE, plot=TRUE, t.pval=0.005)
isat(yy, sis=FALSE, iis=FALSE, tis=TRUE, plot=TRUE, t.pval=0.005)
isat(yy, sis=TRUE, iis=TRUE, tis=TRUE, plot=TRUE, t.pval=0.005)
```

Méthode GETS

Date: Tue Dec 10 15:04:02 2019
Dependent var.: training_dlibaseSWTI
Method: Ordinary Least Squares (OLS)
Variance-Covariance: White (1980)
No. of observations (mean eq.): 262
Sample: 2 to 263

Mean equation:

	coef	std.error	t-stat	p-value
mconst	-0.0042890	0.0093211	-0.4601	0.6458364
ari	-0.1804697	0.0572684	-3.1513	0.0018331
Business_Tendency_Surveys	8.7187327	2.2942783	3.8002	0.0001835
Capacity_Utilization	0.0290142	0.7989594	0.0363	0.9710616
Consumer_Sentiment	-0.0756529	0.0961574	-0.7868	0.4322011
CPI	15.6953173	2.0454538	7.6733	4.247e-13
GISS_Temperature	-0.0196638	0.0220311	-0.8925	0.3729980
GEPU_current	0.0840644	0.2738424	0.3070	0.7591253
GEPU_ppp	-0.0560090	0.2520630	-0.2222	0.8243461
Gold_Princing_Index	-0.2500812	0.1053187	-2.3745	0.0183625
Indice_Kilian	-0.0271219	0.0102416	-2.6482	0.0086307
Industrial_Production	2.7610587	1.0059888	2.7446	0.0065180
Industrial_Capacity	-1.8038162	2.1982279	-0.8206	0.4127047
M2	-2.4744897	1.4166979	-1.7467	0.0819810
OECD_6NME_Industrial_Production	0.6030362	0.7839684	0.7692	0.4425283
Petroleum_and_other_liquids_stocks_US	-1.1126805	0.8033421	-1.3851	0.1673242
Spread	0.0151602	0.0236673	0.6406	0.5224256
Trade_Weighted_USD	-2.0337959	2.2271300	-0.9132	0.3620622
Taux_change_effectif_USD	-0.1776255	0.8387449	-0.2118	0.8324628
US_Ending_Stocks_Crude_Oil	-1.6084443	0.8299263	-1.9381	0.0537941
Three_Component_Index	0.0112011	0.0695457	0.1611	0.8721810
News_Based_Policy_Uncert_Index	-0.0132614	0.0500331	-0.2651	0.7911970
VIX	-0.0455807	0.0270535	-1.6848	0.0933258

Diagnostics and fit:

	Chi-sq	df	p-value
Ljung-Box AR(2)	4.0029426	2	0.13514
Ljung-Box ARCH(1)	0.0010539	1	0.97410
SE of regression	0.06501		
R-squared	0.48761		
Log-lik.(n=262)	355.84340		

SPECIFIC mean equation:

	coef	std.error	t-stat	p-value
ar1	-0.1827574	0.0561817	-3.2530	0.0012975
Business_Tendency_Surveys	9.0742797	2.1261473	4.2679	2.791e-05
CPI	16.9708294	1.6820998	10.0891	< 2.2e-16
Gold_Pricing_Index	-0.2337898	0.0981420	-2.3822	0.0179512
Indice_Kilian	-0.0297369	0.0098876	-3.0075	0.0028996
Industrial_Production	2.9055965	0.9702800	2.9946	0.0030206
M2	-3.6326626	0.8225526	-4.4163	1.489e-05
Taux_change_effectif_USD	-1.1463421	0.3831485	-2.9919	0.0030465
US_Ending_Stocks_Crude_Oil	-2.4763913	0.6603651	-3.7500	0.0002192

Diagnostics and fit:

	Chi-sq	df	p-value
Ljung-Box AR(2)	3.805047	2	0.14919
Ljung-Box ARCH(1)	0.065622	1	0.79782

SE of regression	0.06422
R-squared	0.47062
Log-lik. (n=262)	352.02703

SPECIFIC mean equation:

	coef	std.error	t-stat	p-value
Business_Tendency_Surveys	8.980314	2.210169	4.0632	6.443e-05
CPI	12.430945	1.620882	7.6692	3.613e-13
Indice_Kilian	-0.024697	0.010412	-2.3719	0.018436
Industrial_Production	2.490258	0.991654	2.5112	0.012648
M2	-2.930898	0.868264	-3.3756	0.000851
Petroleum_and_other_liquids_stocks_US	-2.167092	0.656647	-3.3002	0.001103
Trade_Weighted_USD	-0.712309	0.285244	-2.4972	0.013147

Diagnostics and fit:

	Chi-sq	df	p-value
Ljung-Box AR(1)	0.28571	1	0.59298
Ljung-Box ARCH(1)	0.31107	1	0.57703

SE of regression	0.06561
R-squared	0.44551
Log-lik. (n=263)	346.72776

Le package **gets** propose également la **méthode de saturation d'indicateurs** (IS, *Indicator Saturation*) pour la détection des ruptures (*breaks*) et d'outliers proposée par Hendry, Johansen, and Santos (2008) avec la fonction `isat`

Cette méthode IS permet 3 approches :

- **Saturation d'indicateurs d'impulsion** (IIS, *impulse indicator saturation*)

$$y_t = \mu + \sum_{j=2}^n \delta_j \mathbf{1}_{\{t \geq j\}} + u_t$$

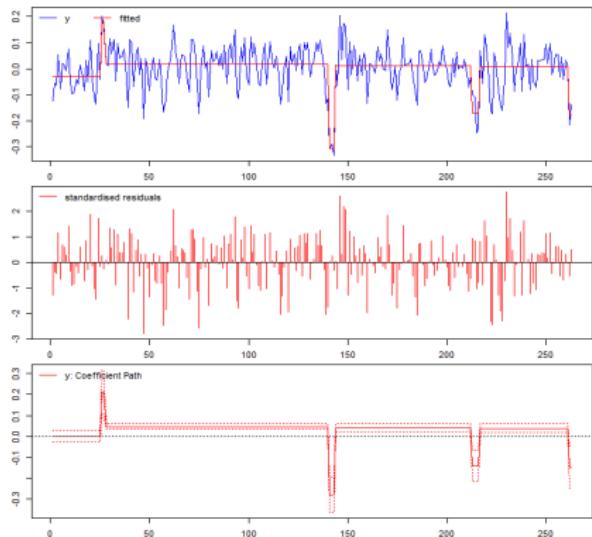
- **Saturation d'indicateurs d'étape** (SIS, *step-indicator saturation*)

$$y_t = \mu + \sum_{j=1}^n \delta_j \mathbf{1}_{\{t=j\}} + u_t$$

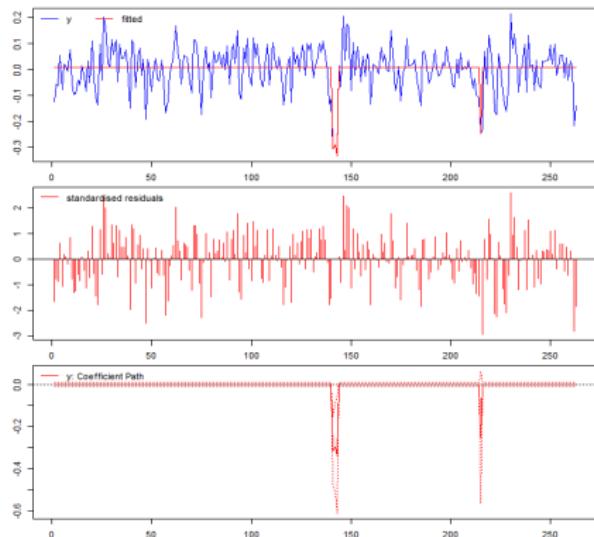
- **Saturation d'indicateurs de tendance** (TIS, *trend-indicator saturation*)

$$y_t = \mu + \sum_{j=1}^n \delta_j \mathbf{1}_{\{t>j\}} + u_t$$

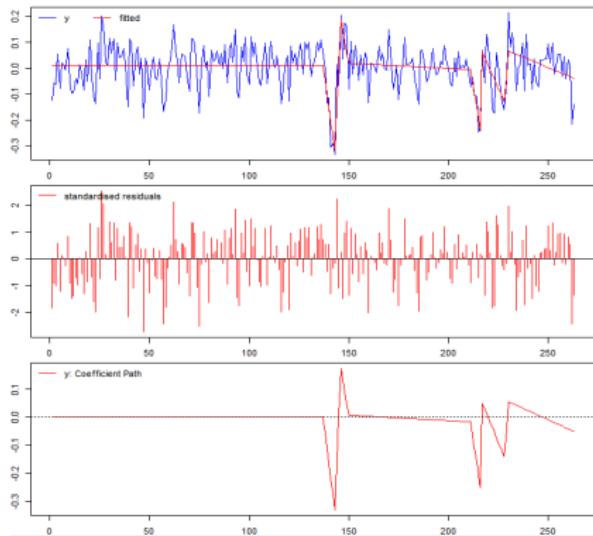
avec n le nombre d'observations, et $u_t \sim i.i.d. \mathcal{N}(0; u^2)$



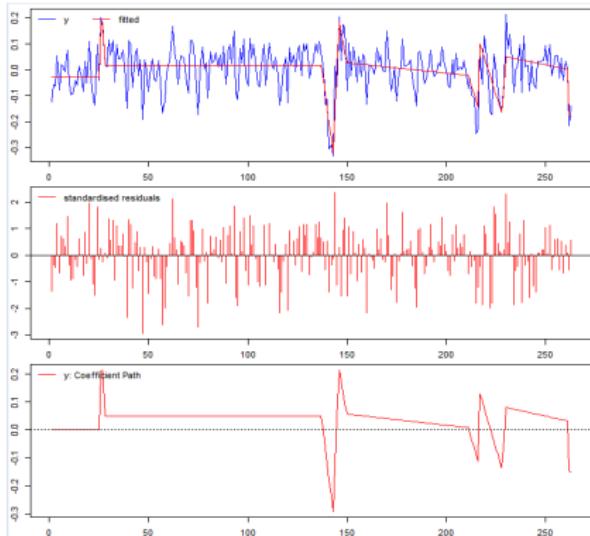
(a) SIS indicator



(b) IIS indicator



(a) TIS indicator



(b) All indicators

Approche "Statistique"

Machine Learning

Machine Learning

Avant de proposer une définition concrète du **Machine Learning** (ML), il convient de le différencier du **Deep Learning** (DL)

Pour pouvoir être performant :

- le DL nécessite des bases de **données massives** pour obtenir un **bon apprentissage**
- tant que le modèle pourra s'**entraîner**, les résultats n'en seront que **meilleurs**
- les résultats de ces données n'ont pas nécessairement à être numériques, classifiés ou sous forme de score (images, sons ...)
- l'algorithme sous-jacent du DL est finalement un **réseau de neurones** qui tente de comprendre en s'aidant des similitudes au sein de son immense base de données qui lui a permis d'**apprendre**

A l'inverse, le **Machine Learning**

- nécessite des bases de données moins importantes
- se fonde sur des modèles dont l'objectif est l'optimisation et d'éviter le sur-apprentissage
- les résultats de ces modèles sont numériques (quantitatif), classifiés (qualitatif) ou sous forme de score (binaire)

L'**algorithme d'apprentissage** du ML peut se présenter sous deux formes :

- **ML Supervisé** (*Supervised Learning*) : il utilise les données des variables explicatives X tout en prenant en compte leur **influence** sur la variable expliquer Y
- **ML Non-supervisé** (*Unsupervised Learning*) : il tient compte **uniquement** des variables explicatives X dans son acheminement

Machine Learning

On peut également faire une classification entre les différentes méthodes de ML :

- Méthodes d'ensemble ou apprentissage ensembliste
 - Bagging
 - Boosting / XGBoosting
 - Random forest
- Méthodes paramétriques
 - Régressions pénalisées
 - Régressions sur composantes principales
 - Analyse en composantes principales parcimonieuses

Régressions pénalisées

Rappels des régressions MCO

La **Méthode des Moindres Carrés Ordinaires (MCO)** (OLS, *Ordinary Least Square*) est basée sur la résolution du problème de minimisation

$$\hat{\beta}_{mco} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \sum_{i=1}^N \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2$$

$$\hat{\beta}_{mco} = \underset{\beta}{\operatorname{argmin}} \|\mathbf{Y} - \mathbf{X}\beta\|^2 \quad \text{où } \|\cdot\| \text{ norme Euclidienne}$$

Propriétés de la régression MCO

- l'estimateur MCO $\hat{\beta}_{mco}$ n'est pas défini si la matrice des régresseurs n'est pas de rang plein sinon pas de solution unique pour $\hat{\beta}_{mco}$. **Rang plein :**

$$\text{rang}(\mathbf{X}'\mathbf{X}) = p+1 \quad \Rightarrow \quad \mathbf{X}'\mathbf{X} \text{ inversible}$$

- l'estimateur MCO $\hat{\beta}_{mco}$ est **linéaire**

$$\hat{\mathbf{Y}} = \mathbf{X}\hat{\beta}_{mco} = \mathbf{H}_{mco}\mathbf{Y} \quad \text{avec } \mathbf{H}_{mco} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' \quad (\mathbf{H}_{mco} : \text{hat matrix})$$

- l'estimateur MCO $\hat{\beta}_{mco}$ est **non biaisé** : $E(\hat{\beta}_{mco}) = \beta$
- l'estimateur MCO est de **variance minimale** : la variance la plus faible parmi tous les estimateurs sans biais linéaires possibles (meilleure précision)

$$\text{Var}(\hat{\beta}_{mco}) = \sigma^2(\mathbf{X}'\mathbf{X})^{-1} = \text{MSE}(\hat{\beta}_{mco}) \quad \text{avec } \hat{\sigma}^2 = \frac{1}{N-p-1} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

- il existe une **forme analytique** de l'estimateur MCO $\hat{\beta}_{mco}$

$$\hat{\beta}_{mco} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$$

Deux problèmes importants peuvent apparaître lorsque l'on traite de données à grande dimension lors de la régression MCO :

1. Le rang

Dans le modèle de régression linéaire, l'utilisation des MCO suppose que la matrice des régresseurs $\mathbf{X}'\mathbf{X}$ soit de **rang plein** et donc **inversible**

Lorsque le nombre de variables dépasse le nombre d'observations ($p \gg N$) alors la matrice des régresseurs n'est pas de rang plein

$\Rightarrow \mathbf{X}'\mathbf{X}$ non inversible

Solution

- Régressions pénalisées : Régression LASSO

2. La colinéarité

Dans le modèle de régression linéaire, l'utilisation des MCO ou du maximum de vraisemblance (MV) suppose que les **variables explicatives** X_j ($j = 1, \dots, p$) sont **linéairement indépendantes**

Colinéarité parfaite (ou stricte) : si les variables sont reliées entre elles par des variables de type linéaire : cas rare (ex. désaisonnalisation par variables dichotomiques \Rightarrow *dummy variable trap*)

Quasi-colinéarité (effet de redondance) : lorsque des relations entre variables dont le coefficient de corrélation est proche de 1 ($\rho \neq 1$) \Rightarrow biaise les estimateurs MCO ou MV

- les estimateurs tendent à être très grands en valeur absolue (corrélation $\Rightarrow \det(X'X) \approx 0$)
- les variance-covariances des estimateurs tendent à être très grandes
- la variance totale des $\hat{\beta}$ devient grande

Détection de la colinéarité : cette détection peut être réalisée par des ratios ou par des tests

- le calcul du R^2
- le test de Klein (1962)
- le test VIF (*Variance Inflation Factor*) de Marquardt (1960)
- le test de Farrar et Glauber (1967)
- le test de Haitovsky (1969)
- le test de Belsley, Kuh et Welsch (1980)

Solutions de la colinéarité

- Régression sur composantes principales
- Régressions pénalisées : Régression Ridge

Le compromis biais-variance

Pour évaluer la performance d'un prédicteur ou d'un estimateur le critère le plus souvent utilisé est l'**erreur quadratique moyenne** (MSE, *Mean Square Error*)

Hastie et alii (2009) proposent la décomposition de la MSE en 3 termes

$$\begin{aligned} MSE = E[(y^* - \hat{y}^*)^2] &= \sigma^2 + (E[\hat{y}^*] - y^*)^2 + E[(\hat{y}^* - E[\hat{y}^*])^2] \\ &= \sigma^2 + \text{biais}^2 + \text{Variance} \end{aligned}$$

- \hat{y}^* est la prédiction, et y^* la vraie valeur
- σ^2 : **terme d'erreur incompressible** : variance de y
- $(E[\hat{y}^*] - y^*)^2$: **biais**² : insuffisances intrinsèques du modèle (variables manquantes ...)
- $E[(\hat{y}^* - E[\hat{y}^*])^2]$: **variance estimée** : instabilité du modèle (précision)

La modélisation explicative va rechercher à **minimiser le biais** afin d'obtenir la représentation la plus précise (ajustement)

- les critères d'évaluation impliquent alors la **significativité statistique** et les **mesures de type R^2** ou les **critères d'information**
- Risque : on peut améliorer le R^2 en rendant le modèle plus complexe (ex. ajout de variables) car il diminuera le biais mais augmentera la variance
⇒ possible signe de **sur-apprentissage**

La modélisation prédictive va rechercher à **minimiser la combinaison biais-variance** (*bias-variance trade-off*)

- la complexité d'un modèle **diminuera son biais** mais **augmentera sa variance**
- le biais est décroissant de la complexité, et la variance croissante
- préférable de sélectionner la variable que d'en rajouter

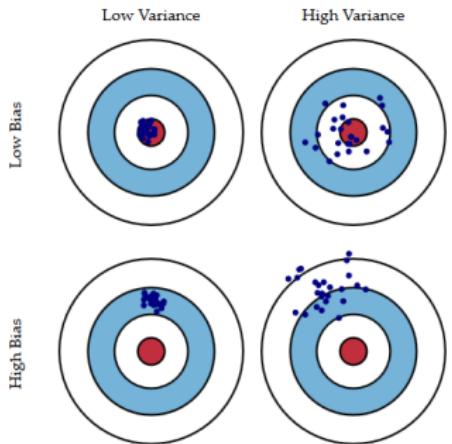


Fig. 1 Graphical illustration of bias and variance.

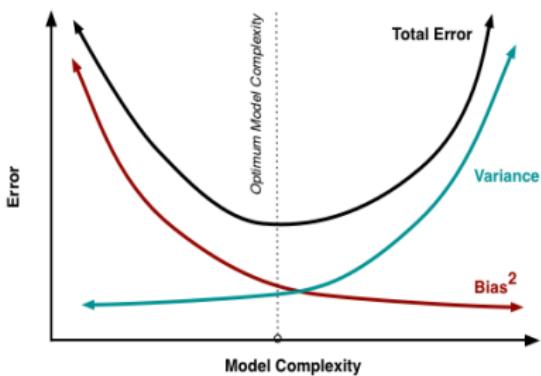


Fig. 6 Bias and variance contributing to total error.

Principe de la régularisation

- **Objectif** : éviter le sur-apprentissage, cad apprendre de l'échantillon de données d'apprentissage, mais pas trop ... (pas de sur dépendance)
- **Principe** : accepter une légère augmentation du biais (**estimateur biaisé**) pour obtenir une réduction plus importante de la variance (**variance plus petite**)
- **Procédure** : diriger (réguler) la modélisation en imposant des **contraintes** sur les paramètres estimés de la régression

Au final, le modèle ainsi obtenu devrait être plus **performant** puisque l'on diminue l'erreur de prédiction espérée (MSE)

Préambule

- *shrinkage* : rétrécissement : on rétrécit les plages de valeurs que peuvent prendre les paramètres estimés $\hat{\beta}$
- les **variables explicatives X** sont **standardisées** (centrées et réduites)
- la **variable expliquée Y** doit être **centrée** pour évacuer la constante de la régression ($\bar{Y} = 0$)
- Ces transformations évitent les problèmes d'échelle des variables lors de l'estimation des paramètres : non invariante à l'échelle (*not scale invariant*)
- Comme on est dans un cadre de régression linéaire il faut que les variables **Y** et **X** soient **stationnaires**
⇒ régressions falacieuses (*spurious regressions*, Granger et Newbold, 1974, [\[pdf\]](#))

Les méthodes de shrinkage

Les **méthodes de shrinkage** (rétrécissement) consistent à réduire le nombre de variables à retenir pour la construction d'un modèle.

Il existe certaines méthodes qui ajoutent une **pénalisation** (ou **régularisation**) sur les estimateurs : **régressions régularisées** ou **pénalisées**

- la **régression Ridge**
- la **méthode LASSO**
- la **méthode Elastic-Net**
- ...

Ces méthodes sont basées sur du *backward* par minimisation de la variance des estimateurs.

Les estimateurs de ces méthodes pénalisées sont des **estimateurs biaisés** puisqu'une contrainte y est ajoutée, au contraire des estimateurs BLUE (Best Linear Unbiased Estimators) lors d'une régression MCO

L'estimation des coefficients d'une **régression pénalisée** (ou régularisée) est définie par

$$\hat{\beta} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 \quad \text{sous la contrainte } \sum_{j=1}^p \|\beta_j\|^q \leq \tau$$

On peut réécrire la régression pénalisée de la manière suivante (forme Lagrangienne)

$$\hat{\beta} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^p \|\beta_j\|^q \right\}$$

où $\|\cdot\|^q$ représente la **norme ℓ_q**

- si $q = 1 \Rightarrow$ norme $\ell_1 \Rightarrow \|\beta_j\|^q = |\beta_j| \Rightarrow$ **régression LASSO**
- si $q = 2 \Rightarrow$ norme $\ell_2 \Rightarrow \|\beta_j\|^q = \beta_j^2 \Rightarrow$ **régression Ridge**

Il y a une relation inverse entre τ ($\tau \geq 0$) et λ ($\lambda \geq 0$), **paramètre de régularisation** ou de **pénalité** (*tuning parameter*)

- si $\tau \rightarrow 0 \quad \Leftrightarrow \quad \lambda \rightarrow +\infty \quad \Rightarrow \quad \hat{\beta}_{ridge} = \hat{\beta}_{lasso} = 0$
- si $\tau \rightarrow +\infty \quad \Leftrightarrow \quad \lambda \rightarrow 0 \quad \Rightarrow \quad \hat{\beta}_{ridge} = \hat{\beta}_{lasso} = \hat{\beta}_{mco}$
- plus λ est élevé, plus la régularisation est forte
- lorsque $\lambda \uparrow \Rightarrow$ le biais de l'estimateur $\hat{\beta}_{Ridge}$ a tendance à augmenter et sa variance à diminuer \Rightarrow compromis à trouver ?

Régression Ridge

La **régression Ridge**, proposée par Hoerl et Kennard (1970), consiste à ajouter des contraintes sur les estimateurs en réduisant leur amplitude et par conséquent leur sur-influence sur les prédictions.

L'idée du Ridge est donc non pas de faire un régression linéaire classique mais une **régression régularisée ou pénalisée** qui rend très faible certains coefficients de l'estimation pour les variables semblant être les moins pertinentes
⇒ **rétrécit (shrink) l'estimateur OLS vers zéro** (sans les annuler)

Propriétés de la régression Ridge

- l'estimateur Ridge $\hat{\beta}_{ridge}$ est bien défini même si la matrice des régresseurs n'est pas de rang plein
 $(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})$ toujours inversible \Rightarrow solution unique pour $\hat{\beta}_{ridge}$

- l'estimateur Ridge $\hat{\beta}_{ridge}$ est linéaire

$$\hat{\mathbf{Y}} = \mathbf{H}_{ridge(\lambda)} \mathbf{Y} \quad \text{avec } \mathbf{H}_{ridge} = \mathbf{X}(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'$$

- l'estimateur Ridge $\hat{\beta}_{ridge}$ est biaisé :

$$E(\hat{\beta}_{ridge}) = -\lambda(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\beta \neq \beta$$

- l'estimateur Ridge $\hat{\beta}_{ridge}$ est de variance plus faible que l'estimateur des moindres carrés lorsque celui-ci existe

$$Var(\hat{\beta}_{ridge}) = \sigma^2 \mathbf{W}' \mathbf{X} \mathbf{W} < Var(\hat{\beta}_{mco}) \quad \text{avec } \mathbf{W} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}$$

- il existe une forme analytique de l'estimateur Ridge $\hat{\beta}_{ridge}$

$$\hat{\beta}_{ridge} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{Y}$$

La **régression Ridge** a pour objectif de déterminer une estimation de rétrécissement des β vers 0 pour les variables explicatives les moins pertinentes via les **paramètres ridge** $\lambda_1, \dots, \lambda_p$, avec $\lambda_j \in \mathbb{R}_+ = \{\lambda \in \mathbb{R} | \lambda \geq 0\}$, $j = 1, \dots, p$, et p le nombre de variables explicatives.

Comme les estimations des β dépendent de $\lambda_1, \dots, \lambda_p$ il est important d'**optimiser** ces paramètres ridge $\lambda_1, \dots, \lambda_p$: sélection de la valeur du paramètre de régularisation

Une méthode d'optimisation est basée sur la **minimisation d'un critère de sélection de modèle** (MSC, *model selection criterion*) :

- les **critères d'information** (IC, *information criteria*)
- les **critères de validation croisée** (CV, *cross-validation*)

Avantage et inconvénient

Avantage : les effets de variables explicatives très corrélées (risque de multicolinéarité) se combinent pour se renforcer mutuellement (coef. similaires)

Désavantage : toutes les variables incluses se retrouvent dans le modèle final, pas moyen de dire quelles variables sont les plus importantes

```
# Ridge regression
library(glmnet)
library(doParallel)          # execute foreach loops in parallel with %dopar% operator

# 10-folds CV to choose lambda
# seq(-3, 5, length.out = 100) : random sequence of 100 numbers between -3 and 5
lambda_to_try <- 10^seq(-3, 5, length.out = 100)

# alpha = 0, implementation of ridge regression
# choix du meilleur lambda parmi 100
ridge_cv <- cv.glmnet(x, y, alpha = 0, lambda = lambda_to_try, standardize = T, nfolds = 10)

plot(ridge_cv)              # Figures of lambdas

# Best lambda obtained from CV (lambda.min) - other possibility: lambda.1se
lambda_cv <- ridge_cv$lambda.min

# Evaluation of the final model with the selected lambda
model_cv <- glmnet(x, y, alpha = 0, lambda = lambda_cv, standardize = T)
summary(model_cv)

model_cv$beta                # Ridge betas
```

Régression LASSO

La **méthode du LASSO** (*Least Absolute Shrinkage and Selection Operator*), proposée par Tibshirani (1996), a pour objectif d'**éliminer les variables inutiles** (principe de parcimonie) et uniquement celles-ci.

L'idée du LASSO est donc non pas de faire un régression linéaire classique mais une **régression régularisée ou pénalisée** qui rend nuls certains coefficients de l'estimation (*shrinking the coefficients β toward zero*)

Propriétés de la régression LASSO

- la norme $\ell_1 \Rightarrow$ solutions non linéaires des $\hat{\beta}_{lasso}$ en \mathbf{Y}
 \Rightarrow il n'existe pas de forme analytique de l'estimateur $\hat{\beta}_{lasso}$

- le calcul de l'estimateur $\hat{\beta}_{lasso}$ se fait **numériquement** par des algorithmes d'optimisation convexe
 - **Forward Stagewise Algorithm** : algorithme plus simple et plus facile à implémenter que LARS, avec des résultats très similaires
 - **Algorithme** (itératif) d'apprentissage **LARS** (*Least-Angle Regression*) (Efron et alii, 2004)
 - on débute l'estimation avec tous les $\beta_j = 0$
 - on fait évoluer les coefficients de manière sélective
 - on obtient ainsi des scénarios de solutions (*LASSO path*) à mesure que $\|\beta\|_1$ augmente
 - **Algorithme du gradient (ou de descente de gradient)** : cf. autre section

Algorithm 3.2 Least Angle Regression.

1. Standardize the predictors to have mean zero and unit norm. Start with the residual $\mathbf{r} = \mathbf{y} - \bar{\mathbf{y}}$, $\beta_1, \beta_2, \dots, \beta_p = 0$.
 2. Find the predictor \mathbf{x}_j most correlated with \mathbf{r} .
 3. Move β_j from 0 towards its least-squares coefficient $\langle \mathbf{x}_j, \mathbf{r} \rangle$, until some other competitor \mathbf{x}_k has as much correlation with the current residual as does \mathbf{x}_j .
 4. Move β_j and β_k in the direction defined by their joint least squares coefficient of the current residual on $(\mathbf{x}_j, \mathbf{x}_k)$, until some other competitor \mathbf{x}_l has as much correlation with the current residual.
 5. Continue in this way until all p predictors have been entered. After $\min(N - 1, p)$ steps, we arrive at the full least-squares solution.
-

Algorithm 3.2a Least Angle Regression: Lasso Modification.

- 4a. If a non-zero coefficient hits zero, drop its variable from the active set of variables and recompute the current joint least squares direction.
-

Source: Hastie et alii (2017), *Elements of Statistical Learning*, p.74-76.

Avantage et inconvénients

Avantage : les effets des variables explicatives peu importants sont estimés à 0 \Rightarrow sélection de variables \Rightarrow solution parcimonieuse (*sparse*)

Désavantages :

- en présence de variables explicatives corrélées, la régression LASSO en choisit une arbitrairement et met les autres à 0
- problème lorsque les variables sont fortement corrélées
- dans les problèmes à très grandes dimensions ($p \gg N$), LASSO ne sélectionne que N variables prédictives au maximum (limite de l'algorithme)
- non invariant à l'échelle (*not scale invariant*) \Rightarrow pré-traitement des données (centrées et réduites)

La figure présente les erreurs et la contrainte des régressions Lasso et Ridge simplifiées avec uniquement deux estimateurs pour une meilleure compréhension

- l'estimateur $\hat{\beta}_{1,Lasso} = 0$ car cette variable est moins importante que la 2ème
- l'estimateur $\hat{\beta}_{1,Ridge} \approx 0$ car cette variable semble moins importante que la 2ème

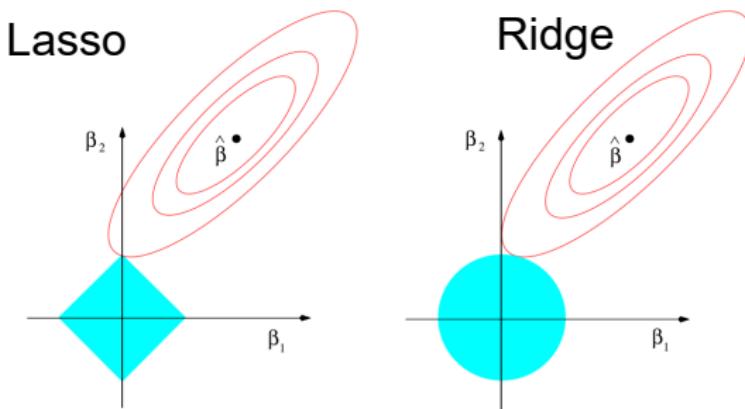


FIGURE 3.11. Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions $|\beta_1| + |\beta_2| \leq t$ and $\beta_1^2 + \beta_2^2 \leq t^2$, respectively, while the red ellipses are the contours of the least squares error function.

```
# LASSO regression
library(glmnet)
library(doParallel)          # execute foreach loops in parallel with %dopar% operator

# 10-folds CV to choose lambda
lambdas_to_try <- 10^seq(-3, 5, length.out = 100)

# alpha = 1, implementation of Lasso regression
lasso_cv <- cv.glmnet(x, y, alpha = 1, lambda = lambdas_to_try, standardize = T, nfolds = 10)

plot(lasso_cv)               # Figures of lambdas

# Best lambda obtained from CV (lambda.1se) - other possibility: lambda.min
lambda_cv <- lasso_cv$lambda.1se

# Evaluation of the final model with the selected lambda
model_cv <- glmnet(x, y, alpha = 1, lambda = lambda_cv, standardize = T)

model_cv$beta                 # Lasso betas
which(! coef(model_cv) == 0, arr.ind = TRUE)    # Get the name of relevant variables
```

Pour montrer la cohérence de la sélection du modèle, la régression LASSO nécessite que la condition irreprésentabilité (IRC, *irrepresentable condition*) soit satisfaite (Zhao et Yu, 2006) :

- L'IRC impose certaines restrictions sur la structure de corrélation entre les variables pertinentes et non pertinentes.
- En d'autres termes, la corrélation entre les deux groupes est limitée et doit être faible.

Cependant, l'IRC peut ne pas toujours tenir dans la pratique et il est difficile, voire impossible, à vérifier.

Néanmoins, même si l'IRC n'est pas satisfaite, la régression LASSO a toujours la propriété de filtrage variable (*variable screening property*), à savoir, la régression LASSO sélectionne les variables pertinentes avec une probabilité élevée, mais elle peut également sélectionner des variables supplémentaires.

En effet, le LASSO a tendance à conserver des variables non associées directement à la réponse (variable à expliquer) mais qui permettent d'améliorer la prédiction. Cependant en terme d'interprétation, il peut être souhaitable de sélectionner uniquement les variables pertinentes (dans un objectif d'explication/identification plutôt que de prédiction).

Par ailleurs, la régression LASSO a des propriétés indésirables lorsque $N \gg p$ ou lorsqu'il existe un groupe de variables parmi lesquelles toutes les corrélations par paire sont très élevées (Zou et Hastie, 2005) ⇒ **risque de biais de sélection entre des variables fortement corrélées.**

- Elastic-Net
- Adaptive LASSO
- Double LASSO
- Relaxed LASSO
- ...

Régression Elastic-net

Pour corriger les possibles erreurs survenues avec la régression LASSO avec les variables corrélées, il existe la régression **Elastic-Net** qui ajoute une pénalisation Ridge en plus de celle de LASSO

Tout comme la **régression Ridge** elle permet de moins discriminer les variables trop corrélées qui étaient néanmoins importante dans la conception du modèle final : partage des poids des groupes de variables prédictives corrélées

Tout comme la **régression LASSO**, les variables non pertinentes auront un coefficient nul et seront écartées pour la construction des modèles (sélection de variables)

Il a un **effet de regroupement** (*grouping effect*) : s'il existe un ensemble de variables parmi lesquelles les corrélations par paire sont élevées, alors Elastic-Net regroupe les variables corrélées.

Zhou et Hastie (2005) proposent la **régression Elastic-Net** en introduisant une pénalité qui est un compromis entre celles des régressions Ridge et LASSO :

$$\begin{aligned}\hat{\beta}_{\text{elastic}-\text{net}} &= \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^p (\alpha \beta_j^2 + (1-\alpha)|\beta_j|) \right\} \\ &= \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda_1 \sum_{j=1}^p \beta_j^2 + \lambda_2 \sum_{j=1}^p |\beta_j| \right\}\end{aligned}$$

- $0 \leq \alpha = \frac{\lambda_2}{\lambda_1 + \lambda_2} \leq 1$
- si $\alpha = 0$ ($\lambda_1 = 0$) \Rightarrow régression LASSO
- si $\alpha = 1$ ($\lambda_2 = 0$) \Rightarrow régression Ridge
- si $\lambda = \lambda_1 = \lambda_2 = 0$ \Rightarrow régression MCO
- des méthodes numériques sont utilisées pour calculer les estimations $\hat{\beta}_{\text{elastic}-\text{net}}$
- le "coefficient-path" selon λ dépend de la valeur de α (pré-déterminée)

```
# Elastic-Net regression
library(glmnet)

# with 0 < alpha < 1      Choose alpha sequencially
a <- seq(0.1, 0.9, 0.05)
search <- foreach(i = a, .combine = rbind) %dopar% {
  cv <- glmnet::cv.glmnet(x, y, alpha = i, lambda = lambdas_to_try, standardize = T, nfolds = 10)
  data.frame(cvm = cv$cvm[cv$lambda == cv$lambda.1se], lambda.1se = cv$lambda.1se, alpha = i)
}

plot(cv)      # Figures of lambdas

# Implementation of EN regression
elasticnet_cv <- search[search$cvm == min(search$cvm), ]

# Best lambda obtained from CV (lambda.1se) - other possibility: lambda.min
lambda_cv <- elasticnet_cv$lambda.1se

# Evaluation of the final model with the selected lambda
model_cv <- glmnet(x, y, lambda = elasticnet_cv$lambda.1se, alpha = elasticnet_cv$alpha)

model_cv$beta                                # EN betas
which(! coef(model_cv) == 0, arr.ind = TRUE)  # Get the name of relevant variables
```

Régression Bridge

La **régression Bridge**, proposée par Frank et Firedman (1993), généralise les régressions Ridge et LASSO en utilisant la norme L_q

$$\hat{\beta}_{bridge} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 \quad \text{sous la contrainte } \sum_{j=1}^p \|\beta_j\|^q \leq \tau$$

On peut réécrire la régression pénalisée sous forme Lagrangienne

$$\hat{\beta}_{bridge} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^p \|\beta_j\|^q \right\}$$

- $q > 0$: paramètre de rétrécissement (*shrinkage parameter*)
- $\lambda > 0$: paramètre de réglage (*tuning parameter*)
- si $0 < q \leq 1 \Rightarrow$ modèles parcimonieux (sélection de variables)
- si $q > 1 \Rightarrow$ rétrécissement des coefficients
- si $q = 2 \Rightarrow$ Ridge régression
- si $q = 1 \Rightarrow$ Lasso régression

```
# Bridge regression
library(rbridge)

# 10-folds CV to choose lambda
# seq(-3, 5, length.out = 100) : random sequence of 100 numbers between -3 and 5
lambda_to_try <- 10^seq(-3, 5, length.out = 100)

# choice of the best lambda among 100
bridge_cv <- cv.bridge(x, y, q=0.5, lambda = lambda_to_try, nfolds = 10)

# Figures of lambdas
plot(bridge_cv)

# Best lambda obtained from CV (lambda.min) - other possibility: lambda.1se
lambda_cv <- bridge_cv$lambda.min

# Evaluation of the final model with the selected lambda
model_cv <- bridge(x, y, q=0.5, lambda = lambda_cv)
summary(model_cv)

model_cv$beta          # Ridge betas
```

Régressions SCAD et MCP

Lorsque p est grand et le nombre de variables pertinentes est petit ($p \gg N$), Fan et Li (2001) et Zhang (2010) ont remarqué que :

- Pour écarter les variables parasites, le paramètre de lissage λ du problème Lasso doit être grand \Rightarrow un biais dans les variables retenues
- La diminution de λ pour réduire le biais \Rightarrow introduction de beaucoup de variables parasites

Pour résoudre ces problèmes :

- Fan et Li (2001) proposent la **pénalité SCAD**
- Zhang (2010) propose la **pénalité MCP**

Régression SCAD

Fan et Li (2001) proposent une fonction de pénalité non-concave avec la **pénalité SCAD** (*smoothly clipped absolute deviation*).

L'estimateur SCAD est donné par :

$$\hat{\beta}_{scad} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \sum_{j=1}^p p_{\lambda}^{scad}(\|\beta_j\|) \right\}$$

où $p_{\lambda}^{scad}(\beta)$ est la fonction de pénalité définie sur $[0, \infty)$, avec $a > 2$, par

$$p_{\lambda}^{scad}(\beta) = \begin{cases} \lambda |\beta| & \text{si } |\beta| \leq \lambda \\ \frac{(2a\lambda|\beta|) - \beta^2 - \lambda^2}{2(a-1)} & \text{si } \lambda < |\beta| \leq a\lambda \\ \frac{\lambda^2(a^2+1)}{2} & \text{si } |\beta| > a\lambda \end{cases}$$

La définition de cette pénalité $p_{\lambda}^{scad}(\beta)$ a les avantages suivants :

- Pour des valeurs élevées de β , cad $|\beta| > a\lambda$, alors la pénalité SCAD est constante par rapport à β .
Autrement dit, une fois que β devient suffisamment grand, les valeurs plus élevées de β ne sont plus pénalisées
- Par contre, la pénalité LASSO est croissante monotone par rapport à β : $p^{lasso}(\beta) = |\beta|$. Cela signifie que pour les valeurs de coefficient élevées, leurs estimations LASSO seront biaisées à la baisse
- Pour de petites valeurs de β , cad $|\beta| \leq \lambda$, la pénalité SCAD est linéaire en β
- Pour des valeurs moyennes de β , cad $\lambda < |\beta| \leq a\lambda$, la pénalité SCAD est quadratique

Cette pénalité permet donc de :

- Pénaliser plus sévèrement que le LASSO les petits coefficients \Rightarrow modèles plus parcimonieux
- Pénaliser moins les grands coefficients \Rightarrow réduction du biais

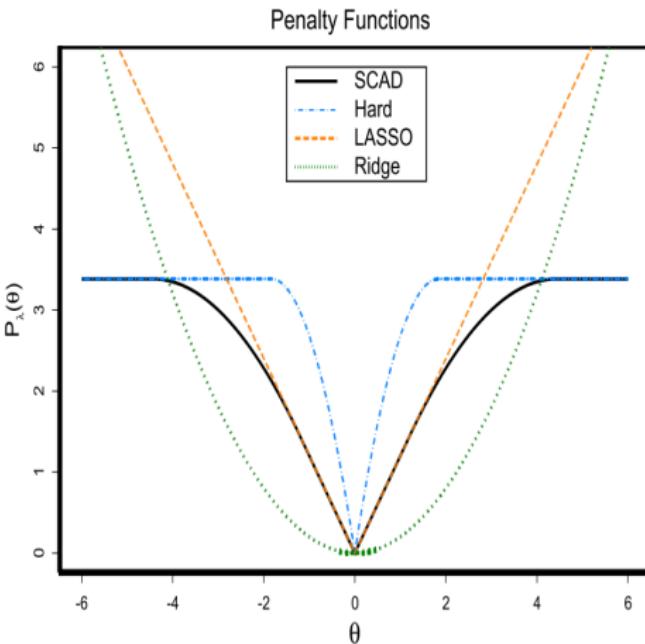


Fig. 1. The four penalty functions of the ridge, hard threshold, LASSO and SCAD techniques.

Hard-thresholding penalty: pénalité à seuil strict

Soft-thresholding penalty: pénalité à souple

Régression MCP

Zhang (2010) propose la **pénalité MCP** (*minimax concave penalty*)

Pour $\lambda \geq 1$ et $a > 1$, considérons la fonction de pénalité $p_\lambda^{mcp}(\beta)$ définie sur $[0, \infty)$ par

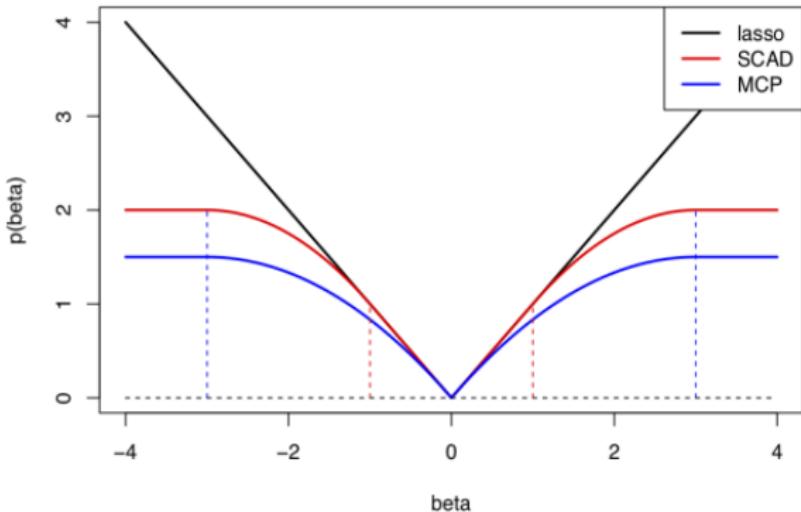
$$p_\lambda^{mcp}(\beta) = \begin{cases} \lambda\beta - \frac{\beta^2}{2a} & \text{si } \beta \leq \lambda a \\ \frac{1}{2}a\lambda^2 & \text{si } \beta > \lambda a \end{cases}$$

L'estimateur MCP est donné par :

$$\hat{\beta}_{mcp} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \sum_{j=1}^p p_\lambda^{mcp}(\|\beta_j\|) \right\}$$

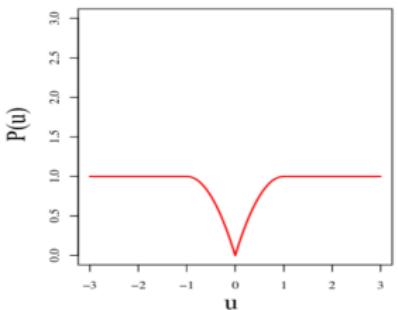
La pénalité MCP commence en appliquant le même taux de pénalisation que le LASSO, mais elle le relaxe d'une manière continue, jusqu'à ce que $t > \lambda\gamma$, où le taux de pénalisation devient nul.

Lasso, SCAD and MCP penalties

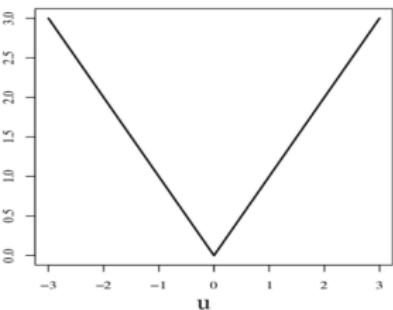


En pointillé rouge : $\beta \in [-\lambda; \lambda] \Rightarrow$ les pénalités SCAD et LASSO sont les mêmes

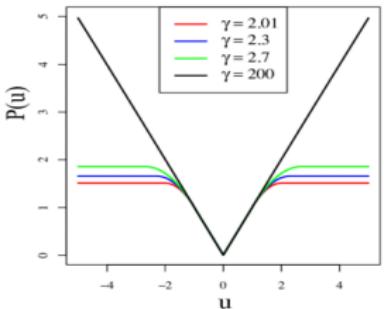
En pointillé bleu : $\beta \notin [-\lambda; \lambda] \Rightarrow$ les pénalités SCAD et MCP sont constantes



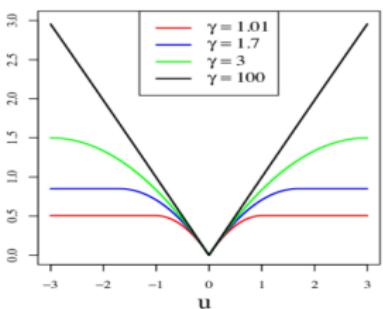
(a) Hard-thresholding Penalty



(b) Soft-thresholding Penalty



(c) SCAD Penalty

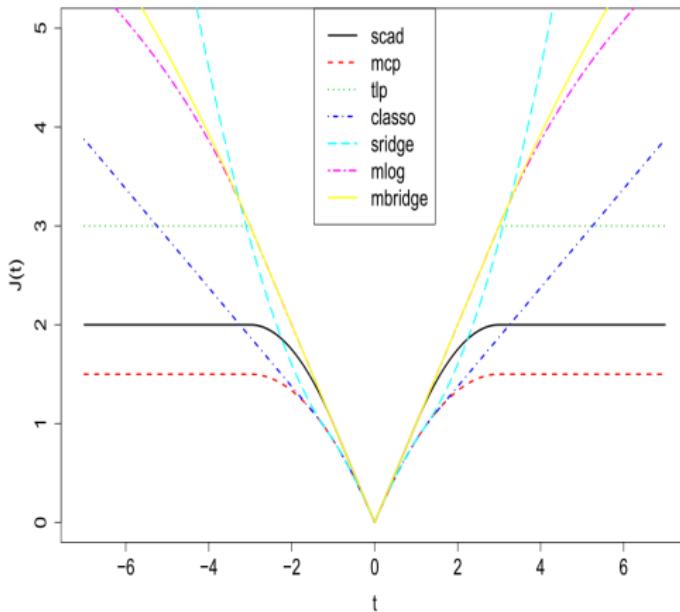


(d) MCP Penalty

Hard-thresholding penalty: pénalité à seuil strict
 $\lambda = 1$ et $\gamma(a)$ varie pour SCAD et MCP

Soft-thresholding penalty: pénalité à souple

Le package `ncpen` de Kim et alii (2018) propose un algorithme uniifié pour l'estimation de pénalités non-convexe



Les pénalités sont calculées pour $\lambda = 1$ et $\gamma = 0.5$

```
# SCAD
library(ncvreg)

fit_SCAD=ncvreg(x, y, penalty = c("SCAD"))
plot(fit_SCAD)
summary(fit_SCAD, lambda=0.10)

# Cross-validation the best lambda
cvfit_SCAD=cv.ncvreg(x, y, penalty = c("SCAD"))
plot(cvfit_SCAD)

# Take the best lambda
lambda_SCAD <- cvfit_SCAD$lambda.min

# Final model
SCAD_Final=ncvreg(x, y, lambda=lambda_SCAD, alpha = 1)
SCAD_Final$beta

which(! coef(SCAD_Final) == 0, arr.ind = TRUE)
```

Régression Adaptive LASSO

Zhou (2006) propose l'**Adaptive LASSO** (aLASSO) pour résoudre le problème de sélection des variables du LASSO,

Il montre que l'inclusion de quelques informations supplémentaires concernant l'importance de chaque variable pourrait considérablement améliorer les résultats

aLASSO utilise la même pénalité que le LASSO avec l'inclusion d'un paramètre de pondération qui provient d'un modèle de première étape qui peut être LASSO, Ridge ou même OLS :

$$\hat{\beta}_{\text{lasso}} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^p \hat{\omega}_j |\beta_j| \right\}$$

- $\hat{\omega}_j = 1/|\hat{\beta}_j^*|^\gamma$, avec $\gamma > 0$
- $\hat{\beta}$: estimation du paramètre initial en 1ère étape (ex. OLS, ridge, LASSO, EN)

Régression Adaptive LASSO

```
# Adaptive Lasso regression
library(glmnet)

#LASSO
model_cv <- glmnet(x, y, alpha = 1, lambda = lambda_cv, standardize = T)
coef_lasso<-predict(model_cv,type="coef",s=lambda_cv)

# Weighted with gamma = 0.5
gamma=0.5
w0<-1/(abs(coef_lasso) + (1/length(y)))
poids.lasso<-w0^(gamma)

# Adaptive LASSO
fit_adalasso <- glmnet(x, y, penalty.factor =poids.lasso)
fit_cv_adalasso<-cv.glmnet(x, y,penalty.factor=poids.lasso)

plot(fit_cv_adalasso)          # Figure of lambdas

# Best lambda obtained from CV (lambda.1se) - other possibility: lambda.min
lambda_cv <- fit_cv_adalasso$lambda.1se

# Evaluation of the final model with the selected lambda
model_cv <- glmnet(x, y, alpha = 1, lambda = lambda_cv, standardize = T)

model_cv$beta                      # Lasso betas
which(! coef(model_cv) == 0, arr.ind = TRUE) # Get the name of relevant variables
```

Régression Weighted fusion

Daye et Jeng (2009) proposent la méthode *Weighted fusion* pour les données corrélées et utile lorsque $p > N$

$$\hat{\beta}_{Wfusion} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^p |\beta_j| + \frac{\mu}{p} \sum_{j=1}^{p-1} \sum_{j>i} \omega_{i,j} (\beta_i - s_{ij} \beta_j)^2 \right\}$$

- $\omega_{i,j} = \frac{|\rho_{i,j}|^\gamma}{1-|\rho_{i,j}|}$
- $\rho_{i,j} = \mathbf{x}_i^T \mathbf{x}_j$: corrélation empirique entre les variables \mathbf{x}_i et \mathbf{x}_j
- $s_{ij} = sign(\rho_{i,j})$: le signe de $\rho_{i,j}$
- $\lambda \geq 0, \mu \geq 0, \gamma \geq 0$: paramètres de lissage

Daye et Jeng (2009) montrent que cette équation peut se réécrire en augmentant les données (quantité \mathbf{Q}) de la manière suivante :

$$\hat{\beta}_{Wfusion} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^p |\beta_j| + \mu \beta^T \mathbf{Q} \beta \right\}$$

$$\mathbf{Q} = \begin{pmatrix} \sum_{k \neq 1} w_{1,k} & -s_{1,2}w_{1,2} & \cdots & -s_{1,p}w_{1,p} \\ -s_{1,2}w_{1,2} & \sum_{k \neq 2} w_{2,k} & \cdots & \vdots \\ \vdots & \vdots & \ddots & -s_{p-1,p}w_{p-1,p} \\ -s_{1,p}w_{1,p} & \cdots & -s_{p-1,p}w_{p-1,p} & \sum_{k \neq p} w_{p,k} \end{pmatrix}$$

avec \mathbf{Q} matrice symétrique et semi-définie positive.

Comme \mathbf{Q} est semi-définie positive alors, par sa décomposition de Cholesky, on a

$$\mathbf{Q} = \mathbf{R}^T \mathbf{R}$$

avec \mathbf{R} est une matrice triangulaire inférieure

On peut alors reformuler l'estimation *weighted fusion* comme

$$\hat{\beta}_{Wfusion} = \operatorname{argmin}_{\beta_0, \dots, \beta_p} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i^* - \sum_{j=1}^p \beta_j x_{i,j}^* \right)^2 + \lambda \sum_{j=1}^p |\beta_j| \right\}$$

avec $y^* = \begin{pmatrix} y \\ 0 \end{pmatrix}$ et $x^* = \begin{pmatrix} x \\ \sqrt{\mu} \mathbf{R} \end{pmatrix}$.

Cela permet le calcul de l'estimateur *weighted fusion* via des algorithmes efficaces disponibles pour le LASSO.

Régression Relaxed LASSO

Dans le même principe d'une procédure en 2 étapes comme pour l'aLASSO, Meinshausen (2007) propose le Régression Relaxed LASSO (reLASSO) :

- Step 1** Estimer une régression LASSO pour prédire la variable à expliquer y_t avec les variables explicatives $x_{j,t}$, avec $j = 1, \dots, p$, et conserver les variables dont les coefficient estimés sont différents de zéro :

$$y_t = \hat{\beta}_0^{lasso} + \hat{\beta}_1^{lasso} x_{1,t} + \cdots + \hat{\beta}_k^{lasso} x_{k,t}$$

L'ensemble des variables retenues par l'estimateur LASSO $\hat{\beta}^{lasso}$ est noté \mathcal{M}_k :

$$\mathcal{M}_k = \{1 \leq k \leq p | \hat{\beta}_k^{lasso} \neq 0\}$$

- Step 2** Estimer une régression LASSO pour prédire la variable à expliquer y_t avec les variables explicatives sélectionnées \mathcal{M}_k

$$\hat{\beta}^{rllasso} = \arg \min_{\beta_k \in \mathcal{M}_k} \sum_{i=1}^N \left(y_i - \sum_{j=1}^k \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^k |\beta_j|$$

À partir de résultats théoriques et numériques, Meinshausen (2007) démontre que le reLASSO produit des modèles **plus parcimonieux** avec une perte de prédiction égale ou inférieure à l'estimateur LASSO pour les données de grande dimension.

Les avantages du reLASSO par rapport au LASSO dans le cadre de grande dimension sont doubles :

- **Estimations parcimonieuses** : le nombre de coefficients sélectionnés est en général beaucoup plus petit pour le reLASSO, sans compromettre la précision des prédictions. Les modèles produits par reLASSO se prêtent donc davantage à l'interprétation.
- **Prédictions plus précises** :
 - si le rapport signal/bruit est très faible, la précision prédictive du LASSO et du reLASSO est comparable
 - Pour un rapport signal/bruit élevé, le reLASSO réalise des prédictions souvent beaucoup plus précises
 - Pour un rapport signal/bruit élevé, le LASSO sélectionnera un plus grand nombre de variables (notamment dites *noise*)

Régression LASSO+Ridge

Liu et Yu (2013) proposent également une procédure en 2 étapes de la manière suivante :

Step 1 Appliquer la régression LASSO pour sélectionner les prédicteurs :

$$\hat{\beta}^{lasso} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^p |\beta_j|$$

On obtient ainsi un ensemble de variables sélectionnées avec des coefficients différents de zéro $\hat{S} = \{j \in \{1, 2, \dots, p\} : \hat{\beta}_j^{lasso} \neq 0\}$.

Step 2 Appliquer la régression Ridge aux variables sélectionnées dans \hat{S}

$$\tilde{\beta}^{lasso+ridge} = \underset{\beta: \beta_j=0, j \in \hat{S}}{\operatorname{argmin}} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{j=1}^p \beta_j^2$$

Multi-step adaptive EN

Xiao et Xu (2015) proposent une approche itérative appelée Multi-step adaptive EN (MSAEN).

Au lieu d'utiliser un (comme dans Ridge et LASSO) ou deux (comme dans EN) régularisation ils développent une procédure de plusieurs étapes itératives où chaque étape utilise des paramètre de régularisation séparés.

Le package [msaenet](#) de Xiao (2019) permet d'estimer

- adaptive EN (aEN) et mutli-step aEN
- adaptive MCP-Net et mutli-step adaptive MCP-Net
- adaptive SCAD-Net et mutli-step adaptive SCAD-Net

La procédure MSAEN est la suivante :

Step 1 Initialisation des poids adaptatifs $\omega_j \equiv 1$ ($j = 1, 2, \dots, p$)

Step 2 Pour $k = 1, 2, \dots, M$ on utilise l'estimation de l'**adaptive EN** (Zou et Zang, 2009) avec la fonction de poids suivante :

$$\lambda_2^{*(k)} \parallel \beta \parallel_2^2 + \lambda_1^{*(k)} \sum_{i=1}^p \omega_j^{k-1} |\beta_j|$$

Step 3 Mise à jour des poids adaptatifs pour $j = 1, 2, \dots, p$:

$$\omega_j^{(k)} = \frac{1}{|\hat{\beta}^{(k-1)}(\lambda_1^{*(k-1)})_j|}$$

- Pour $k = 1$ (*one-stage*) le MSAEN est égale à l'estimation **EN classique** : MSAEN = EN
- Pour $k = 2$ (*two-stage*) le MSAEN correspond à l'estimation **adaptive EN** : MSAEN = aEN

```
library(tidyverse)
library(msaenet)
library(glmnet)

a <- seq(0.1, 0.9, 0.05)
msaenet.fit <- msaenet(x, y, family = "gaussian", init = "enet", alphas = a, tune = "cv", nfolds = 10, seed = 1001)
print(msaenet.fit)
coef(msaenet.fit)
msaenet.fit$beta
which(! coef(msaenet.fit) == 0, arr.ind = TRUE)           # Get the name of relevant variables

msaenet.nzv(msaenet.fit)          # Get Indices of Non-Zero Variables
msaenet.fp(msaenet.fit, 1:5)      # Get the Number of False Positive Selections
msaenet.tp(msaenet.fit, 1:5)      # Get the Number of True Positive Selections

# init = "enet" for EN
# init = "mnet" for MCP
# init = "snet" for SCAD
```

Régressions pénalisées et packages R

- La fonction `lm.ridge` du package **MASS** implémente la régression Ridge
On peut obtenir le critère GCV pour chaque λ : `fit$GCV`
- Le package **glmnet** permet d'implémenter les régressions Ridge, LASSO, Elastic-Net (version 6.2 de R)
- Le package **elasticnet** permet d'implémenter la régression Elastic-Net
- Le package **lars** implémente l'algorithme LARS
- Le package **rbridge** implémente la régression Bridge
- Le package **ncvreg** implémente les régressions SCAD et MCP
- Les packages **lqa**, **parcor**, **lassogrp** permettent d'estimer la régression aLASSO

Variantes de la régression LASSO

- **Group LASSO** (Yuan et Lin, 2007) : permet une sélection de groupes de variables (*Group sparsity*)

$$\hat{\beta}_{grp-lasso} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{s=1}^L \sqrt{\beta_s' K_s \beta_s} \right\}$$

où $s = 1, \dots, L$, avec L le nombre de groupes de variables pré-définis par des sous-matrices \mathbf{X}_s de \mathbf{X}

R packages : `lassogrp`, `gglasoo`

- **Sparse-Group LASSO** (Simon et alii, 2013)

$$\hat{\beta}_{sparse-lasso} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \lambda \sum_{s=1}^L \sqrt{\beta_s' K_s \beta_s} + \mu |\beta| \right\}$$

où $\mu > 0$

R packages : `SGL`

Variantes de la régression LASSO

- Adaptive group LASSO de Wang et Leng (2008)
- Bayesian LASSO de Park et Casella (2008)
- Bayesian Adaptive LASSO de Leng et alii (2014)

Variantes de la régression Elastic-Net

- Adaptive Elastic Net de Zou et Zhang (2009) : combine quadratique régularisation et réticissement Lasso pondérées de manière adaptative

Variantes de la régression Bridge

- Group Bridge de Park et Yoon (2011)
- Bayesian Bridge de Mallick et Yi (2018)

Variante de la régression Ridge

- Adaptive Ridge (Grandvalet, 1998) : chaque coefficient β_j à sa propre pénalisation

$$\hat{\beta}_{a-ridge} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p \beta_j x_{i,j} \right)^2 + \sum_{j=1}^p \lambda_j \beta_j^2 \right\}$$

sous la contrainte $\frac{1}{p} \sum_{j=1}^p \frac{1}{\lambda_j} = \frac{1}{\lambda}$ avec $\lambda_j > 0$

Grandvalet montre que l'Adaptive Ridge et LASSO sont équivalents dans le sens qu'ils donnent la même estimation.

La différence porte sur la contrainte de l'expression de la régression pénalisée sous forme Lagrangienne

- LASSO : $\sum_{j=1}^p |\beta_j| \leq \tau$
- Adaptive Ridge : $\left(\sum_{j=1}^p |\beta_j|^2 \right) / p \leq \tau^2$

Versions robustes aux outliers des régressions pénalisées

- Robuste LARS (RLARS) de Khan et al. (2007)
- LAD-LASSO de Wang et al. (2007) (LAD, *least absolute deviation*)
- WLAD-LASSO de Arslan et al. (2012) (WLAD, *weighted LAD*)
- Sparse LTS de Alfrons et al. (2013) (LTS, *least trimmed square*)
- Robuste non-négative garotte (R-NNG) de Gijbels et Vrissen (2015)

Table 1. A sampling of generalizations of the lasso

<i>Method</i>	<i>Reference</i>	<i>Detail</i>
Grouped lasso	Yuan and Lin (2007a)	$\Sigma_g \ \beta_g\ _2$
Elastic net	Zou and Hastie (2005)	$\lambda_1 \Sigma \beta_j + \lambda_2 \Sigma \beta_j^2$
Fused lasso	Tibshirani <i>et al.</i> (2005)	$\lambda \Sigma \beta_{j+1} - \beta_j $
Adaptive lasso	Zou (2006)	$\lambda_1 \Sigma w_j \beta_j $
Graphical lasso	Yuan and Lin (2007b); Friedman <i>et al.</i> (2007)	$\text{loglik} + \lambda \Sigma^{-1} _1$
Dantzig selector	Candes and Tao (2007)	$\min\{X^T(y - X\beta) \ \infty\} / \ \beta\ _1 < t$
Near isotonic regularization	Tibshirani <i>et al.</i> (2010)	$\Sigma (\beta_j - \beta_{j+1})_+$
Matrix completion	Candès and Tao (2009); Mazumder <i>et al.</i> (2010)	$\ X - \hat{X}\ ^2 + \lambda \ \hat{X}\ _*$
Compressive sensing	Donoho (2004); Candes (2006)	$\min(\ \beta\ _1) \text{ subject to } y = X\beta$
Multivariate methods	Jolliffe <i>et al.</i> (2003); Witten <i>et al.</i> (2009)	Sparse principal components analysis, linear discriminant analysis and canonical correlation analysis

Source: Tibshirani (2011). Regression shrinkage and selection via the lasso: A retrospective. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*

Non-Negative Garotte

Une autre approche de régressions pénalisées est la [régression non-negative garotte](#) (NNG) proposée par Breiman (1995).

Elle se déroule en 2 étapes :

- (1) [estimation initiale](#) des β_j par les [OLS](#) : $\hat{\beta}_{j,OLS}$
- (2) [rétrécissement](#) (*shrinkage*) ou certains coefficients $\hat{\beta}_{OLS} = 0$ ([sélection](#)) en utilisant les [NNG factors](#) \hat{c}

$$\hat{c} = \underset{c_1, \dots, c_p}{\operatorname{argmin}} \left\{ \frac{1}{2N} \sum_{i=1}^N \left(y_i - \sum_{j=1}^p c_j \hat{\beta}_{j,OLS} x_{i,j} \right)^2 + \lambda \sum_{j=1}^p c_j \right\}$$

sous la contrainte $c_j \geq 0, \quad j = 1, \dots, p$

L'estimateur NNG des coefficients β_j est donné par

$$\hat{\beta}_{j,NNG} = \hat{c}_j \hat{\beta}_{j,OLS}$$

La solution de minimisation des facteurs NNG est donnée par

$$\hat{c}_j = \left(1 - \frac{\lambda}{\hat{\beta}_{j,OLS}^2} \right)_+$$

avec $(.)_+$ la partie positive de $(.)$, et $\lambda > 0$

Remarques sur NNG :

- rétrécit et élimine les prédicteurs
- invariant d'échelle (*scale invariant*)
- capacité prédictive comparable à celle de la régression Ridge
- estimation par les MCO \Rightarrow sensible aux hypothèses des MCO
- il faut que $N > p$
- pas de package R
- Yuan et Lin (2007) proposent d'utiliser l'estimateur Ridge comme estimateur initial
- version robuste aux outliers : R-NNG (Gijbels et Vrissen, 2015)

Algorithme du gradient (ou de descente de gradient)

Parfois la résolution analytique n'est pas possible, parce que le nombre de paramètres est élevé par exemple, ou parce que le calcul serait trop coûteux
⇒ approximation avec une **approche itérative**.

Le modèle de régression linéaire estimé par les MCO revient à résoudre le problème de minimisation

$$\hat{\beta}_{mco} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \mathbf{S} = \underset{\beta_0, \dots, \beta_p}{\operatorname{argmin}} \sum_{i=1}^N \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{i,j} \right)^2$$

Il existe une **forme analytique** de l'estimateur MCO $\hat{\beta}_{mco}$

$$\hat{\beta}_{mco} = (\mathbf{X}' \mathbf{X})^{-1} \mathbf{X}' \mathbf{Y}$$

Cette solution de la minimisation nécessite la manipulation de la matrice $(\mathbf{X}' \mathbf{X})$ de dimension $(p+1 \times p+1)$

Cela devient ingérable dès que p est très élevé (grandes dimensions)

L'**algorithme du gradient** (ou descente de gradient) permet de résoudre des problèmes d'optimisation dans les situations de **très forte dimensionnalité** ($p \gg N$) (volumétrie), notamment pour les régressions linéaires (et pénalisées)

$$\beta^{t+1} = \beta^t - \eta \nabla \mathbf{S}^t$$

- η : le **taux d'apprentissage** (*learning rate*) (trop faible \Rightarrow lenteur de convergence ; trop élevé \Rightarrow oscillations)
- $\nabla \mathbf{S}^t$: le **vecteur gradient** de dimension ($p + 1 \times 1$) composé des dérivées partielles de \mathbf{S} par rapport à chaque paramètre du modèle

$$\nabla \mathbf{S}^t = \begin{cases} \frac{\partial \mathbf{S}^t}{\partial \beta_0} = \frac{1}{N} \sum_{i=1}^N (-1) \left(y_i - \sum_{j=1}^p \beta_j^t x_{ij} \right) \\ \frac{\partial \mathbf{S}^t}{\partial \beta_1} = \frac{1}{N} \sum_{i=1}^N (-x_{i1}) \left(y_i - \sum_{j=1}^p \beta_j^t x_{ij} \right) \\ \dots \\ \frac{\partial \mathbf{S}^t}{\partial \beta_p} = \frac{1}{N} \sum_{i=1}^N (-x_{ip}) \left(y_i - \sum_{j=1}^p \beta_j^t x_{ij} \right) \end{cases}$$

L'intérêt de l'algorithme du gradient est de manipuler un vecteur de dimension ($p + 1 \times 1$) et non plus une matrice de dimension ($p + 1 \times p + 1$)

L'algorithme est répété jusqu'à convergence de l'estimation de $\widehat{\beta}$: nombre d'itérations t fixé, ou différence entre valeurs successives de β_t , ou $\|\nabla \mathbf{S}^t\|$ très petit

Algorithme du gradient

Le problème d'optimisation est le suivant pour les régressions pénalisées :

$$E(\beta) = \frac{1}{N} \sum_{i=1}^N L(y_i, f(x_i)) + \lambda R(\beta)$$

- $L(\cdot)$: fonction de perte confrontant la cible observée et la cible prédite (ex. MSE)
- $f(x_i) = \sum_{j=1}^p \beta_j x_{ij}$
- $R(\beta)$: terme de pénalité
 - Norme ℓ_1 (LASSO)

$$R(\beta) = \frac{1}{2} \sum_{i=1}^p |\beta_j|$$

- Norme ℓ_2 (Ridge)

$$R(\beta) = \frac{1}{2} \sum_{i=1}^p \beta_j^2$$

- Elastic-Net

$$R(\beta) = \frac{\alpha}{2} \sum_{i=1}^p \beta_j^2 + (1 - \alpha) \sum_{i=1}^p |\beta_j|$$

- $L(\cdot)$ est choisie dérivable mais $R(\beta)$ ne peut pas être dérivable \Rightarrow solutions : Proximal gradient, Subgradient, Generalized gradient ...
- R packages : `gradDescent`

Sélection de la valeur du paramètre de régularisation

Les critères d'information

Les **critères d'information** (IC, *Information Criteria*) sont des **critères de sélection de modèles** (MCS, *Model Selection Criteria*) et sont basés sur la divergence de Kullback-Leibler (Kullback et Leibler, 1951).

Le modèle \mathcal{M} qui sera sélectionné est celui qui minimise ces IC :

$$\mathcal{M}_{IC} = \operatorname{argmin}_{\mathcal{M}} IC(N, df)$$

Nishii (1984) propose le **critère d'information généralisé** (GIC, *generalized information criterion*)

$$GIC = -\frac{2}{N} LL + \alpha \frac{df}{N}$$

avec

- df : **degrés de liberté effectifs** du modèle
- N : taille de l'échantillon
- α : fonction de pénalité
- L : vraisemblance maximisée pour le modèle \mathcal{M}
- $LL = \ln L$: fonction de log-vraisemblance maximisée pour le modèle \mathcal{M}
- Pour des modèles Gaussiens : $-2LL = N\ln(SSE)$ avec SSE la somme des carrés des erreurs (*sum square of errors*)

Sélection de la valeur du paramètre de régularisation

1. Le critère d'information d'Akaike (AIC) (Akaike, 1973):

$$AIC = -\frac{2}{N} \text{LogL} + \frac{2}{N} df \quad \text{avec } \alpha = 2$$

2. Lorsque df est grand par rapport au nombre d'observations N , cad si $N/df < 40$, il est recommandé d'utiliser l'AIC corrigé (AICc) (Hurvich et Tsai, 1995)

$$AICc = AIC + \frac{2df(df+1)}{N-df-1}$$

3. Le critère d'information de Schwarz (SIC) ou Bayésien (BIC) (Schwarz, 1978)

$$BIC = -\frac{2}{N} \text{LogL} + \frac{\ln(N)}{N} df \quad \text{avec } \alpha = \ln(N)$$

4. Le critère d'information de Hannan-Quinn (HQ) (Hannan et Quinn, 1979)

$$HQ = -\frac{2}{N} \text{LogL} + 2df \frac{\ln(\ln(N))}{N} \quad \text{avec } \alpha = 2\ln(\ln(N))$$

Il peut être considéré comme un intermédiaire entre le AIC et le BIC

Degrés de liberté (df) des modèles MCO et Ridge

- Pour la **régression MCO**, les degrés de liberté sont définis par

$$df_{mco} = \text{tr}(\mathbf{H}_{mco}) = \text{tr}(\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}') = p$$

- Pour la **régression Ridge**, les degrés de liberté *effectifs* sont définis par

$$df_{ridge} = \text{tr}(\mathbf{H}_{ridge}) = \sum_{i=1}^p \frac{\lambda_i}{\lambda_i + \lambda}$$

avec λ_i les valeurs propres de $(\mathbf{X}'\mathbf{X})$.

df_{ridge} une fonction monotone décroissante en λ , avec $df_{ridge} = p$ quand $\lambda = 0$ (pas de régularisation) et $df_{ridge} \rightarrow 0$ quand $\lambda \rightarrow \infty$

Degrés de liberté (df) des modèles LASSO et Elastic-Net

- Pour la **régression LASSO**, les degrés de liberté sont estimés comme le nombre de coefficients non nuls

$$df_{lasso} = \text{nombre de coefficients non nuls} < p$$

- Pour la **régression Elastic-Net**, les degrés de liberté *effectifs* sont définis par

$$df_{elastic-net} = \text{tr}(\mathbf{H}_{elastic-net}) = \text{tr}(\mathbf{X}_A (\mathbf{A}'_A \mathbf{X}_A + \lambda_2 I)^{-1} \mathbf{X}'_A)$$

avec \mathbf{X}_A l'ensemble des A variables actives

Si $\lambda_2 = 0 \Rightarrow$ régression LASSO

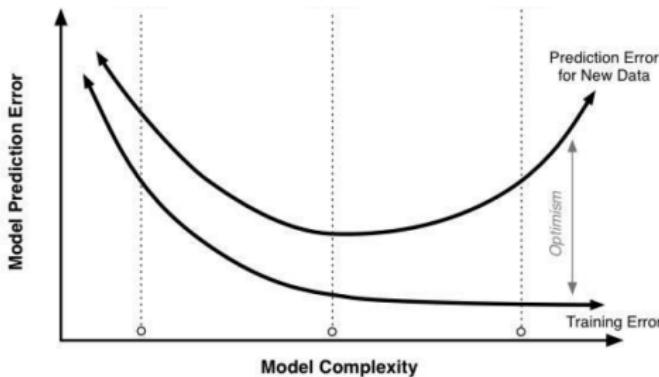
Approche de la performance prédictive

Les critères d'information utilisent deux fois les données :

- une fois pour estimer le modèle (*train*)
- une autre fois pour mesurer la qualité (*test*)
- pas d'utilisation de nouvelles données pour évaluer le modèle

Cela risque d'entrainer un **sur-ajustement** (*over-fitting*) et une mauvaise et imprécise sélection de modèles

$$\text{True Prediction Error} = \text{Training Error} + f(\text{Model Complexity})$$



La validation croisée (*cross-validation*) et ré-échantillonnage (*resampling*)

De manière générale la **validation croisée** se base sur un découpage des données en K blocs (K -*folds*) de tailles approximativement égales :

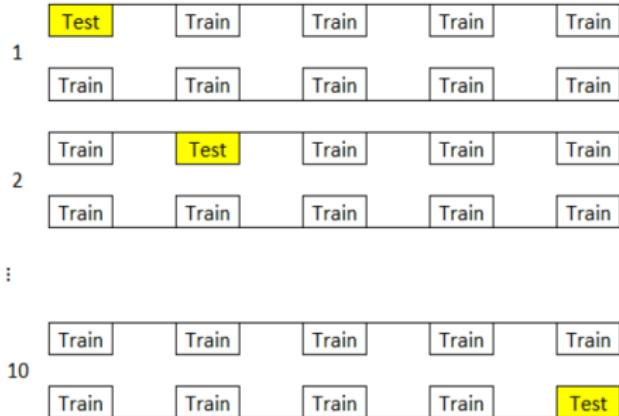
- une partie servant d'**estimation** (*training*)
- une autre partie pour évaluer la **qualité de prédiction** (*validation or test*)

L'algorithme de validation croisée (*K-fold cross-validation* ou *leave-many-out cross-validation*, LMOCV) peut se résumer ainsi :

- estimation du modèle et donc des $\hat{\beta}$ sur les $(K - 1)$ blocs
- évaluation des prédictions issues de $\hat{\beta}$ (risque) par rapport au bloc écarté : erreurs de prédiction
- processus répété pour chaque K
- calcul de l'erreur de prédiction moyenne des K blocs
- les choix standards de $K = 5, 10, N$ (*leave-one-out cross-validation*, LOOCV)

Sélection de la valeur du paramètre de régularisation

FIGURE 1 – Principe de la validation croisée à 10 segments



Pour évaluer l'erreur de prédiction moyenne un critère de la performance prédictive souvent utilisé est le critère de validation croisée généralisée (GCV, *generalized cross-validation*) (Cravan et Wahba, 1979)

$$GCV = \frac{1}{N} \sum_{i=1}^N \frac{(y_i - \hat{y}_i)^2}{(1 - df/N)^2}$$

En petit échantillon on peut utiliser le critère GCV corrigée (GCV_c) (Boonstra et alii, 2015)

$$GCV_c = \frac{1}{N} \sum_{i=1}^N \frac{(y_i - \hat{y}_i)^2}{(1 - df/N - \alpha/N)^2}$$

- $\alpha > 0$: force de la pénalité de complexité du modèle
- si $\alpha = 0 \Rightarrow GCV_c = GCV$

Cas particulier de la validation croisée

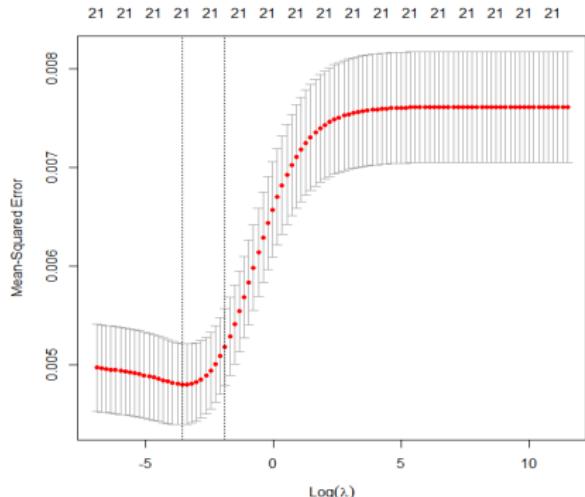
Dans le cas particulier où $K = N$ pour la validation croisée nous obtenons la validation croisée *leave-one-out* (LOOCV)

On utilise alors le **PRESS** (*predicted residual error sum of squares*) :

$$PRESS_{LOOCV} = \sum_{i=1}^N y_i - \hat{y}_i^{-i} = \sum_{i=1}^N \left(\frac{(y_i - \hat{y}_i)}{1 - h_{ii}} \right)^2 = PRESS_{GCV}$$

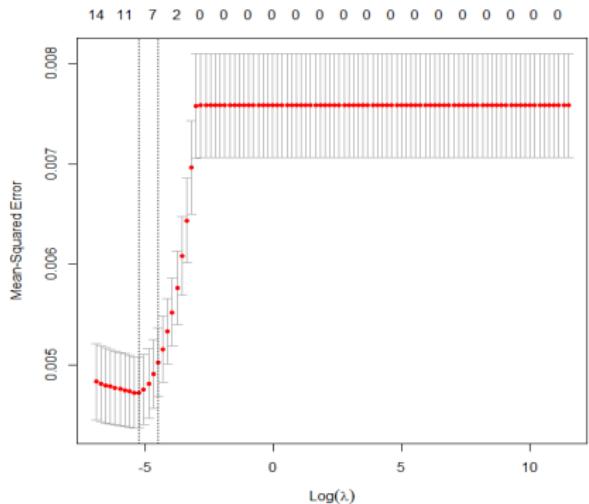
avec

- \hat{y}_i^{-i} : prédiction de l'individu i à partir du modèle construit sur $(n - 1)$ observations
- h_{ii} : éléments de la diagonale de la matrice \mathbf{H}



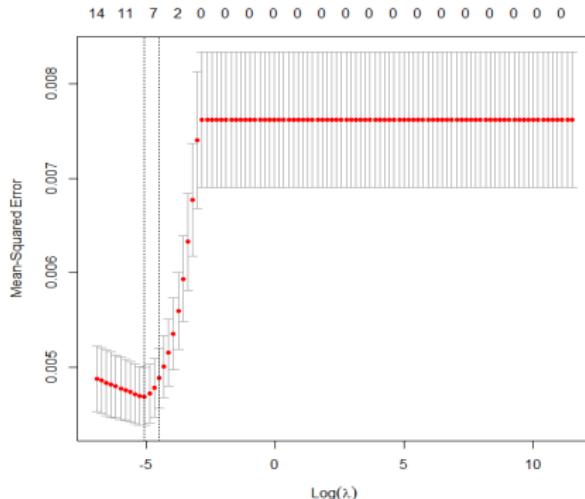
21 x 1 sparse Matrix of class "dgCMatrix"	
	s0
Business_Tendency_Surveys	6.4578298260
Capacity_Utilization	-0.0024815480
Consumer_Sentiment	-0.0377007902
CPI	9.4586562450
GISS_Temperature	-0.0197003456
GEPU_current	-0.0105113244
GEPU_ppp	-0.0041036757
Gold_Princing_Index	-0.0702745338
Indice_Kilian	-0.0014826665
Industrial_Production	0.2402480298
Industrial_Capacity	-0.9322096443
M2	-1.5532556596
OECD_6NME_Industrial_Production	0.7028077778
Petroleum_and_other_liquids_stocks_US	-1.2420925827
Spread	-0.0035742421
Trade_Weighted_USD	-0.4276815033
Taux_change_effectif_USD	-0.1677783482
US_Ending_Stocks_Crude_Oil	-0.9695350682
Three_Component_Index	-0.000998067
News_Based_Policy_Uncert_Index	-0.0005766030
VIX	-0.0441525571

Estimation of $\hat{\beta}_{Ridge}$



21 x 1 sparse Matrix of class "dgCMatrix"	
s0	
Business_Tendency_Surveys	6.03424995
Capacity_Utilization	.
Consumer_Sentiment	.
CPI	10.73864478
GISS_Temperature	.
GEPU_current	.
GEPU_ppp	.
Gold_Frincing_Index	.
Indice_Kilian	.
Industrial_Production	.
Industrial_Capacity	.
M2	-0.09389436
OECD_6NME_Industrial_Production	.
Petroleum_and_other_liquids_stocks_US	-0.66323135
Spread	.
Trade_Weighted_USD	-0.35567818
Taux_change_effectif_USD	.
US_Ending_Stocks_Crude_Oil	-0.09343582
Three_Component_Index	.
News_Based_Policy_Uncert_Index	.
VIX	-0.00545480

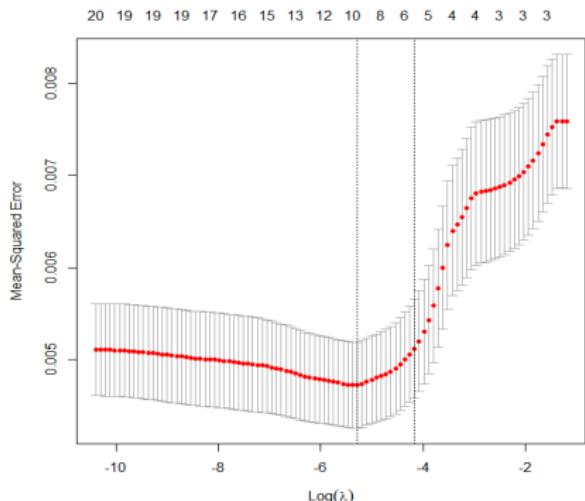
Estimation of $\hat{\beta}_{Lasso}$



Values of λ from CV for Elastic-Net with $\alpha = 0.5$

Estimation of $\hat{\beta}_{\text{elastic-net}}$

	s_0
Business_Tendency_Surveys	6.007389638
Capacity_Utilization	.
Consumer_Sentiment	.
CPI	10.442971617
GISS_Temperature	.
GEPU_current	.
GEPU_ppp	.
Gold_Frincing_Index	.
Indice_Kilian	.
Industrial_Production	.
Industrial_Capacity	.
M2	-0.196900110
OECD_6NME_Industrial_Production	.
Petroleum_and_other_liquids_stocks_US	-0.698997430
Spread	.
Trade_Weighted_USD	-0.377686799
Taux_change_effectif_USD	.
US_Ending_Stocks_Crude_Oil	-0.150903459
Three_Component_Index	.
News_Based_Policy_Uncert_Index	.
VIX	-0.007476581



Values of λ from CV for Adaptive LASSO

Estimation of $\hat{\beta}_{AdapLasso}$

21 x 1 sparse Matrix of class "dgCMatrix"	
s0	
Business_Tendency_Surveys	4.7143870
Capacity_Utilization	.
Consumer_Sentiment	.
CPI	10.2662760
GISS_Temperature	.
GEPU_current	.
GEPU_ppp	.
Gold_Frincing_Index	.
Indice_Kilian	.
Industrial_Production	.
Industrial_Capacity	.
M2	.
OECD_GNME_Industrial_Production	.
Petroleum_and_other_liquids_stocks_US	-0.2097648
Spread	.
Trade_Weighted_USD	-0.1997905
Taux_change_effectif_USD	.
US_Ending_Stocks_Crude_Oil	.
Three_Component_Index	.
News_Based_Policy_Uncert_Index	.
VIX	.

	GETS	Ridge	LASSO	Elastic-Net	AdapLASSO
Business_Tendency_Surveys	8.98	6.458	6.034	6.007	4.714
Capacity_Utilization		-0.002			
Consumer_Sentiment		-0.038			
CPI	12.43	9.46	10.74	10.44	10.27
GISS_Temperature		-0.020			
GEPU_current		-0.011			
GEPU_PPP		-0.004			
Gold_Pricing_Index		-0.070			
Indice_Kilian	-0.024	-0.001			
Industrial_Production	2.49	0.240			
Industrial_Capacity		-0.933			
M2	-2.931	-1.553	-0.094	-0.197	
OECD_6NME_Industrial_Production		0.702			
Petroleum_and_other_liquids_stocks_US	-2.167	-1.242	-0.663	-0.700	-0.210
Spread		-0.004			
Trade_Weighted_USD		-0.428	-0.356	-0.378	-0.200
Taux_change_effectif_USD	-0.712	-0.168			
US_Ending_Stocks_Crude_Oil		-0.970	-0.093	-0.151	
Three_Component_Index		-0.001			
News_Based_Policy_Uncert_Index		-0.001			
VIX		-0.044	-0.006	-0.007	

Comparaison des estimations des β

L'approche SIS

Fan et Lv (2008) proposent une méthode de pré-sélection des variables nommée *Sure independence screening* (SIS) fondée sur un **apprentissage des corrélations** (*correlation learning*).

L'idée de cette approche est que seules les variables dont la corrélation absolue est plus élevée avec la variable dépendante devrait être utilisée dans la modélisation.

Soit $\omega := (\omega_1, \dots, \omega_p)'$ le vecteur de corrélations marginales des variables explicatives avec la variable dépendante Y_t :

$$\omega = X' Y$$

avec X une matrice $(N \times p)$ des variables explicatives standardisées par colonne.

Pour tout $\lambda \in]0; 1[$ on trie les p grandeurs par composante du vecteur ω dans un ordre décroissant et on définit un sous-modèle

$$\mathcal{M}(\lambda) = \{1 \leq i \leq p : |\omega_i| \text{ est parmi les premiers } [\lambda N] \text{ les plus grands de tous}\}$$

avec $[\lambda N]$ la partie entière de λN .

Cette méthode SIS est un moyen simple de réduire le modèle complet $\{1, \dots, p\}$ jusqu'à un sous-modèle $\mathcal{M}(\lambda)$ de taille $d = [\lambda N] < N$

Avantages

- La méthode SIS est invariante à l'échelle des données (variables standardisées)
- La procédure utilise chaque variables x_j indépendamment comme un prédicteur utile pour prévoir Y_t

```
# SIS procedure
library(SIS)

mod01 <- SIS(x, y, family="gaussian", penalty="MCP", tune="cv", nfolds=10, nsis=100)

# indices selected by only SIS
var <- mod01$sis.ix0
show(var)
# indices selected by (I)SIS with regularization step
var2 <- mod01$ix
show(var2)
# coefficients of the final model selected by (I)SIS
coefvar2 <- mod01$coef.est
show(coefvar2)

# family = c("gaussian", "binomial", "poisson", "cox")
# penalty = c("SCAD", "MCP", "lasso")
# nsis = number of predictors recruited by (I)SIS
```

Sélection de variables et de modèles

Il existe d'autres approches plus économétriques pour la **sélection de variables et de modèles** dans le cadre des *big data*, notamment afin de prendre en compte le problème de l'**incertitude du modèle** (*model uncertainty*).

Le problème de l'**incertitude du modèle** peut provenir de

- l'instabilité des paramètres estimés sur une longue période
- le changement dans le temps de la significativité de variables explicatives (expansions vs récessions)

Les méthodes proposées sont :

- *Bayesian model averaging* (BMA) et *Bayesian model selection* (BMS)
- *Dynamic model averaging* (DMA) et *Dynamic model selection* (DMS)

Terminologie

- "MA" : faire la moyenne des différents modèles à chaque point du temps
- "MS" : un seul modèle est sélectionné à chaque point du temps sur la base de ses performances afin de faire une sélection dynamique

Packages R

- BMA et BMS approches : **BMS** package de Zeugner et Feldkircher (2015)
- DMA et DMS approches : **fDMA** package de Drachal (2020) et **eDMA** package de Catania et Nonejad (2018)