1 Développement des mocks

Dans ce chapitre, nous présentons la construction des *mocks* : des spectres de quasars simulés, dont les forêts Lyα et le champ de quasars possèdent les bonnes fonctions d'auto corrélation et de corrélation croisée. Ces mocks visent à reproduire les données d'eBOSS et de DESI. Ils sont nommés `SaclayMocks` et présentés dans ?. Le code est écrit en Python [1] et se trouve en accès libre sur GitHub [2]. L'utilisation de ces mocks et leur validation seront présentés dans les chapitres suivants.

1 Objectifs des mocks

Contrairement à ce qu'on appelle les simulations, les mocks ne contiennent pas de physique à proprement parler : ils ne sont pas utilisés afin de déduire des paramètres astrophysiques. Certaines simulations, les simulations hydrodynamiques, permettent de mesurer des effets astrophysiques, comme par exemple le biais de l'hydrogène ou du Lyα. Mais ces simulations sont très couteuses car elles nécessitent de modéliser les effets physiques qui affectent les paramètres mesurés. Les mocks, quant à eux, sont conçus afin de répliquer fidèlement et rapidement un jeu de données, dans le but de tester l'analyse qui sera appliquée sur ces données. Les mocks sont donc utilisés afin

- de vérifier la mesure des paramètres $\alpha_{\parallel}$, $\alpha_{\perp}$: cette mesure est-elle non biaisée ?
- d'identifier les potentielles systématiques : la présence de métaux et d'HCD dans les données est-elle bien modélisée ? Affecte-t-elle la mesure de $\alpha_{\parallel}$, $\alpha_{\perp}$?
- de tester la matrice de distorsion : la distorsion de la fonction de corrélation due à l'ajustement du continuum du quasar est-elle correctement prise en compte par la matrice de distorsion ?
- de vérifier l'estimation de la matrice de covariance : la matrice de covariance, calculée à partir des données, est-elle bien estimée ?

La production et l'analyse d'un grand nombre de mocks permet de répondre précisément à ces questions. Ces mocks sont donc nécessaires pour pouvoir valider l'analyse menée sur les données.

Les mocks décrits dans ce manuscrit s'inscrivent dans les projets eBOSS et DESI. Ils sont utilisés dans l'analyse Lyα des données complète d'eBOSS (**?**), et seront utilisés dans l'analyse Lyα de DESI. L'objectif de ces mocks est donc de répliquer fidèlement les données Lyα d'eBOSS et de DESI. Ces relevés couvrent un volume de plusieurs dizaines de h^{-1} Gpc^3, et les échelles sondées grâce au Lyα descendent jusqu'à la centaine de h^{-1} kpc. Les mocks nécessitent donc de reproduire un volume immense, avec une bonne résolution. Les simulations dites N-body sont des simulations qui traitent le problème à N corps. Elles sont initialisées avec une distribution de matière noire, représentée par des macro-particules de masse $\sim 10^9 M_{\odot}$, à un redshift élevé ($z > 100$). Puis, à chaque pas de temps, ces macro-particules sont déplacées en considérant uniquement les interactions gravitationnelles. Le champ de matière initial évolue ainsi jusqu'à $z = 0$. Ces simulations sont très utiles pour étudier les effets de la gravité à grande échelle. Cependant elles ne sont pas adaptées à notre utilisation : afin d'avoir la résolution et le volume requis, la simulation nécessiterait beaucoup trop de macro-particules pour être réalisable dans un temps raisonnable.

Les simulations hydrodynamiques fonctionnent de la même manière que les simulations N-body. Elles incluent, en plus des macro-particules de matière noire, la physique baryonique présente dans le milieu galactique. Les baryons sont aussi représentés par des macro-particules. Afin de résoudre l'intérieur des galaxies, les macro-particules utilisées possèdent une masse plus faible que dans le cas des simulations N-body. En contrepartie, le volume simulé est plus petit. Dans le cas des simulations hydrodynamiques, la densité, la pression et la tempéature sont tracées dans chaque cellule. Certains

1. https ://www.python.org/
2. https ://github.com/igmhub/SaclayMocks

effets astrophysiques, comme les supernovae ou les AGN peuvent aussi être présents. Cependant, ces simulations ne sont pas non plus adaptées à notre utilisation car le volume d'univers simulé est bien trop petit : quelques dizaines de h^{-1} Mpc^3.

Ainsi, avoir un grand volume et une grande résolution requiert l'utilisation des *champs aléatoires gaussiens* (GRF pour Gaussian Random Field). Ce sont des champs qui en chaque point prennent une valeur aléatoire selon une statistique gaussienne. Une fois générés, il est possible de donner à ces champs n'importe quelle fonction de corrélation, en utilisant la transformation de Fourier. Les GRF sont donc idéaux pour simuler le champ de matière à grande échelle. Cependant, l'utilisation des GRF ne donne pas accès aux non linéaritées qui peuvent émerger dans l'évolution des similutaions N-body et hydrodynamiques. La seule information provient de la fonction de corrélation que l'on applique au GRF. Mais cela est entièrement suffisant pour l'utilisation que nous en avons dans ce manuscrit.

2 Construction des mocks

Dans cette section, nous détaillons comment les mocks sont générés. Nous présentons d'abord la génération des champs de densité, puis comment les quasars sont tirés à partir de ces champs de densité. Nous expliquons ensuite comment nous calculons la densité le long de la ligne de visée de chaque quasar. Enfin, nous présentons comment la densité le long de la ligne de visée est transformée en fraction de flux transmis, et comment nous tirons les HCD.

2.1 Les champs de densité

La première étape dans la création des mocks est de générer les boîtes qui contiennent le champ de densité δ. D'abord, un GRF est généré dans une boite de $2560 \times 2560 \times 1536$ voxcells, chaque voxcell faisant $d_{cell}^3 = (2{,}19\, h^{-1}\,\mathrm{Mpc})^3$. Afin que le champ δ possède la bonne fonction de corrélation, une transformation de Fourier 3D [1] est appliquée sur la boîte, puis le champ δ_k ainsi obtenu est multiplié par

$$\sqrt{\frac{P(k)}{d_{cell}^3}}\,, \tag{1.1}$$

où $P(k)$ est le spectre de puissance désiré. Ce procédé garanti que le champ δ_k suive le spectre de puissance $P(k)$. Il est ensuite possible d'obtenir le champ δ dans l'espace réel grâce à une transformation de Fourier inverse du champ δ_k.

Le GRF pourrait être tiré directement dans l'espace k. Dans ce cas, il nous faut tirer deux champs gaussiens : un pour la partie réelle, et un autre pour la partie imaginaire. La transformation de Fourier prenant moins de temps que la génération du champ aléatoire, nous préférons générer le champ dans l'espace réel plutôt que dans l'espace k. Dans la suite nous décrivons les différents champs nécessaires à la construction des mocks : les champs utilsés pour tirer les quasars, le champ utilisé pour créer l'absorption Lyα, ainsi que les champs de vitesse et de gradient de vitesse. Afin de garantir leur corrélation, tous ces champs sont construits à partir du même champ initial δ_k.

Les quasars

Afin de construire un relevé de quasars corrélés, nous tirons les quasars selon le champ dans l'espace réel $\delta_{\mathbf{QSO}}$, construit à partir de δ_k. Une première solution est de tirer les quasars dans les cellules dont le champ $\delta_{\mathbf{QSO}}$ est supérieur à un certain seuil.

1. Nous utilisons la librairie pyFFTW (https ://github.com/pyFFTW/pyFFTW), une adaptation python de la librairie FFTW (http ://www.fftw.org/).

Cette solution produit une fonction de corrélation correcte aux grandes échelles, mais pas aux petites. En effet, comme expliqué précédemment, l'utilisation des GRF ne permet pas de capturer l'évolution non linéaire du champ de matière, qui se manifeste aux petites échelles. Plutôt que de modéliser ces non linéarités, nous considérons une seconde solution : nous considérons que les quasars suivent une distribution log-normale. Ceci permet d'obtenir une meilleure corrélation aux petites échelles. Ce choix est souvent fait pour simuler des relevés de galaxies (Agrawal et al., 2017), et est en accord avec ce qui est observé dans les données (Clerkin et al., 2016). Ainsi, dans chaque voxcell, les quasars sont tirés avec une probabilité

$$P \propto \mathrm{e}^{\delta_q} \,, \tag{1.2}$$

où δ_q est le champ de densité dans le voxcell considéré. Comme montré par **?**, afin que les quasars suivent la fonction de corrélation $\xi(r)$, le champ δ_q doit suivre la fonction de corrélation

$$\xi_q(r) = \ln(1+\xi(r)) \,. \tag{1.3}$$

De manière à obtenir un relevé synthétique de quasars dont le biais dépend de z, nous utilisons trois champs qui suivent des distributions log-normales, à des redshifts différents. La probabilité pour tirer les quasars dépend de l'interpolation de ces 3 champs. Ces champs sont construits aux redshits $z_1 = 1{,}9$, $z_2 = 2{,}75$, et $z_3 = 3{,}6$. Pour chacun des champs, nous partons du spectre de puissance de la matière $P_{matière}(k)$ à $z = 0$, fournit par Camb (Lewis et al., 1999). Nous multiplions ensuite ce spectre de puissance par $(b_{\mathrm{QSO}}(z_i)G(z_i))^2$, où $i \in [1,2,3]$. A l'aide de la transformation de Fourier, nous calculons la fonction de corrélation $\xi_i(r)$. Puis, nous déterminons le spectre de puissance $P_{\mathrm{QSO},i}(k)$, à appliquer au champ δ_k, comme la transformée de Fourier de $\xi_{\mathrm{QSO},i}(r) = \ln(1+\xi_i(r))$ (équation **??**). Une fois les trois spectres de puissances $P_{\mathrm{QSO},i}(k)$ obtenus, nous construisons 3 champs

$$\delta_{k,i}(k) = \delta_k(k)\sqrt{\frac{P_{\mathrm{QSO},i}(k)}{V_{cell}}} \,, \tag{1.4}$$

où δ_k est le GRF dans l'espace de Fourier. Une fois ces 3 champs construits, nous appliquons à chacun d'entre eux une transformation de Fourier inverse afin d'obtenir les boîtes $\delta_{\mathrm{QSO},i}$. Ces champs seront interpolés en z, puis les quasars seront ensuite tirés avec une probabilité $\propto \exp(\delta_{\mathrm{QSO}}(z))$, où δ_{QSO} est le champ interpolé. Nous expliquons cette étape dans la section 2.2.

Le champ Lyα

Afin de construire le champ d'absorption Lyα, nous avons besoin du champ de densité de l'hydrogène neutre. Comme expliqué dans la section **??**, la fraction de flux transmis F est reliée à la profondeur obtique τ par

$$F = exp(-\tau) \,. \tag{1.5}$$

De plus, la formule FGPA (Fluctuating Gunn Peterson Approximation) permet de relier la profondeur optique τ au contraste de densité δ à $z = 0$:

$$\tau(z) = a(z)\mathrm{e}^{b(z)G(z)\delta} \tag{1.6}$$

Les paramètres a et b sont des paramètres à ajuster afin d'obtenir le bon biais du Lyα et la bonne transmission moyenne $\overline{F}$. Leur détermination est décrite dans la section **??**. Le facteur de croissance G prend en compte l'évolution avec le redshift du champ de densité δ. Ainsi il nous suffit de construire un GRF qui suit la fonction de corrélation à $z = 0$ pour simuler le champ d'absorption du Lyα. Pour ce faire, nous partons du même champ δ_k utilisé pour construire les 3 champs log-normaux des quasars. Ceci garanti la corrélation croisée entre les quasars et le champ d'absorption Lyα. Le spectre

de puissance de la matière $P_{matière}(k)$ à $z = 0$ est ensuite appliqué au champ δ_k. Enfin, nous obtenons le champ de densité $\delta_{matière}$ à $z = 0$ qui servira au calcul du champ d'absorption du Lyα en effectuant la transformée de Fourier du champ

$$\delta_{k,matière}(\vec{k}) = \delta_k(\vec{k})\sqrt{\frac{P_{matière}(\vec{k})}{V_{cell}}} \,. \tag{1.7}$$

Les champs des vitesses

Afin d'inclure les RSD dans nos mocks, nous simulons aussi le champ des vitesses. A l'ordre linéaire, le champ des vitesses $v_{k,n}$ dans l'espace k selon la direction $\vec{u}_n$, avec $n \in [\mathrm{X}, \mathrm{Y}, \mathrm{Z}]$, est relié au champ de densité δ_k par la relation (#prov mettre la démo ?)

$$v_{k,n}(\vec{k}) = \frac{ik_n}{k^2}\dot{a}f\delta_k(\vec{k}) \,. \tag{1.8}$$

Il est fréquent de considérer que le champ de vitesse des traceurs est le même que celui de la matière sous-jacente. Autrement dit, le champ de vitesse des traceurs est non biaisé. En ce qui concerne les quasars, nous simulons les RSD en déplaçant chaque quasar proportionnellement à sa vitesse le long de la ligne de visée (équation **??**). Dans ce but, nous calculons les trois champs de vitesses $v_{k,\mathrm{X}}$, $v_{k,\mathrm{Y}}$ et $v_{k,\mathrm{Z}}$ à $z = 0$, comme

$$v_{k,n}(\vec{k}) = \frac{-ik_n}{k^2}H_0\frac{dG}{dz}\delta_{k,matière}(\vec{k}) \,; \quad n \in [\mathrm{X}, \mathrm{Y}, \mathrm{Z}] \,. \tag{1.9}$$

Cette équation est équivalente à l'équation 1.8 pour $z = 0$. Comme précédemment, le champ $\delta_{k,matière}$ est le même que celui utilisé pour construire les champs $\delta_{\mathrm{QSO},i}$ des quasars et le champ $\delta_{matière}$ utilisé pour le Lyα, ceci afin de garantir la corrélation entre la densité des traceurs et leurs vitesses particulières. A l'aide d'une transformation de Fourier inverse, nous obtenons les trois champs de vitesse à $z = 0$ dans l'espace réel v_{X}, v_{Y} et v_{Z}.

Concernant le champ d'absorption Lyα, les RSD sont prises en compte par une modification de la formule FGPA. Pour ce faire, nous nécessitons le gradient de vitesse $\eta_{\parallel}$ à $z = 0$. Le gradient η_{nm} selon la direction $\vec{u}_m$ de la vitesse v_n est défini comme

$$\eta_{nm}(\vec{k}) = \frac{k_n k_m}{k^2}f\delta_k(\vec{k}) \,. \tag{1.10}$$

Cette équation permet de retrouver la formule de kaiser :

$$\begin{aligned} \delta_k^s(\vec{k}) &= \delta_k(\vec{k}) + \eta_{\parallel}(\vec{k}) \,, \\ &= (1 + f\mu_k^2)\delta_k(\vec{k}) \,. \end{aligned} \tag{1.11}$$

Nous construisons donc 6 champs de gradients de vitesse, à $z = 0$, comme

$$\eta_{nm}(\vec{k}) = \frac{k_n k_m}{k^2}\delta_k(\vec{k}) \,; \quad (n, m) \in [\mathrm{X}, \mathrm{Y}, \mathrm{Z}]^2 \,. \tag{1.12}$$

Le champ δ_k utilisé est le GRF initial, afin de garantir les corrélations entre les différents champs. Nous omettons volontairement le facteur $f(z = 0)$ à ce stade. Il sera pris en compte lors de l'ajout de la dépendance en redshift (voir section 2.4). A l'aide d'une transformation de Fourier, nous obtenons les 6 champs de gradients de vitesses à $z = 0$ dans l'espace réel η_{XX}, η_{YY}, η_{ZZ}, η_{XY}, η_{YZ} et η_{XZ}.

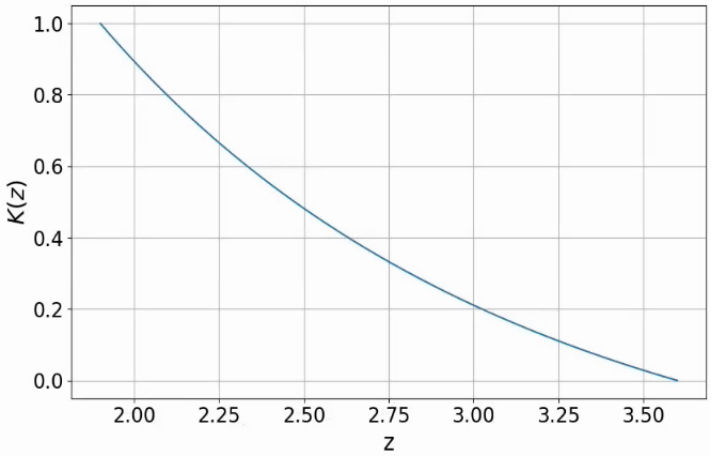

FIGURE 1.1 – Coefficient $K(z)$ défini dans l'équation 1.16.

2.2 Le relevé de quasars

Une fois tous ces champs construits, nous définissons la géométrie du relevé. Les boîtes, d'une taille $2560 \times 2560 \times 1536$ selon les axes X, Y et Z respectivement, sont placées à un redshift central $z_0 = 1.70975268202$, et à une ascension droite α_0 et une déclinaison δ_0 (voir équation 1.17). Leurs dimensions permettent de couvrir les redshifts $1.3 < z < 3.6$. L'observateur est considéré être à $z = 0$, au centre du plan (X, Y). Afin de construire le catalogue de quasars, nous utilisons les trois champs $\delta_{\mathrm{QSO},i}$ construits précédemment, aux redshits $z_1 = 1{,}9$, $z_2 = 2{,}75$, et $z_3 = 3{,}6$. Dans chacun des cas, nous calculons

$$\hat{\delta}_{\mathrm{QSO},i}(z) = \exp\left(\delta_{\mathrm{QSO},i} \frac{b_{\mathrm{QSO}}(z)(1+z_i)}{b_{\mathrm{QSO}}(z_i)(1+z)}\right), \tag{1.13}$$

où b_{QSO} est le biais des quasars. Le redshift dans chaque voxcell est calculé en utilisant l'équation **??**. Les paramètres cosmologiques utilisés sont donnés dans l'équation **??**. Toutes les distances sont comobiles. Une fois les 3 champs $\hat{\delta}_{\mathrm{QSO},i}$ construits, nous construisons les deux champs

$$\hat{\delta}_{\mathrm{QSO},12}(z) = \hat{\delta}_{\mathrm{QSO},1}(z) \frac{z_2 - z}{z_2 - z_1} + \hat{\delta}_{\mathrm{QSO},2}(z) \frac{z - z_1}{z_2 - z_1}, \tag{1.14}$$

$$\hat{\delta}_{\mathrm{QSO},23}(z) = \hat{\delta}_{\mathrm{QSO},2}(z) \frac{z_3 - z}{z_3 - z_2} + \hat{\delta}_{\mathrm{QSO},3}(z) \frac{z - z_2}{z_3 - z_2}, \tag{1.15}$$

puis, nous construisons le champ interpolé

$$\hat{\delta}_{\mathrm{QSO}}(z) = K(z) \left(\hat{\delta}_{\mathrm{QSO},12}(z) - \hat{\delta}_{\mathrm{QSO},23}(z)\right) + \hat{\delta}_{\mathrm{QSO},23}(z), \tag{1.16}$$

où $K(z)$ est un coefficient qui varie entre 0 et 1. Il est représenté sur la figure **??**. Les quasars sont ensuite tirés dans chaque voxcell, avec une probabilité $P \propto \hat{\delta}_{\mathrm{QSO}}$. Pour ce faire, nous générons un champ ϕ aléatoire uniforme entre 0 et 1, et les voxcells pour lesquelles $\phi < N(z)\hat{\delta}_{\mathrm{QSO}}$ hébergent potentiellement un quasar. Le facteur $N(z)$ est un facteur de normalisation. Les quasars dont le redshift est en dehors de l'intervalle $[1{,}8; 3{,}6]$ sont écartés. Les quasars dont l'ascension droite et la déclinaison sont en dehors des intervalles $[-\Delta\alpha; \Delta\alpha]$ et $[-\Delta\delta; \Delta\delta]$ sont aussi écartés. L'ascension droite α et la déclinaison δ du point $(\mathrm{X}, \mathrm{Y}, \mathrm{Z})$ sont définies comme

$$\alpha = \arctan\left(\frac{\cos(\alpha_0)\mathrm{X} - \sin(\delta_0)\sin(\alpha_0)\mathrm{Y} + \cos(\delta_0)\sin(\alpha_0)\mathrm{Z}}{-\sin(\alpha_0)\mathrm{X} - \sin(\delta_0)\cos(\alpha_0)\mathrm{Y} + \cos(\delta_0)\cos(\alpha_0)\mathrm{Z}}\right), \tag{1.17}$$

$$\delta = \arcsin\left(\frac{\cos(\delta_0)\mathrm{Y} + \sin(\delta_0)\mathrm{Z}}{\sqrt{\mathrm{X}^2 + \mathrm{Y}^2 + \mathrm{Z}^2}}\right). \tag{1.18}$$

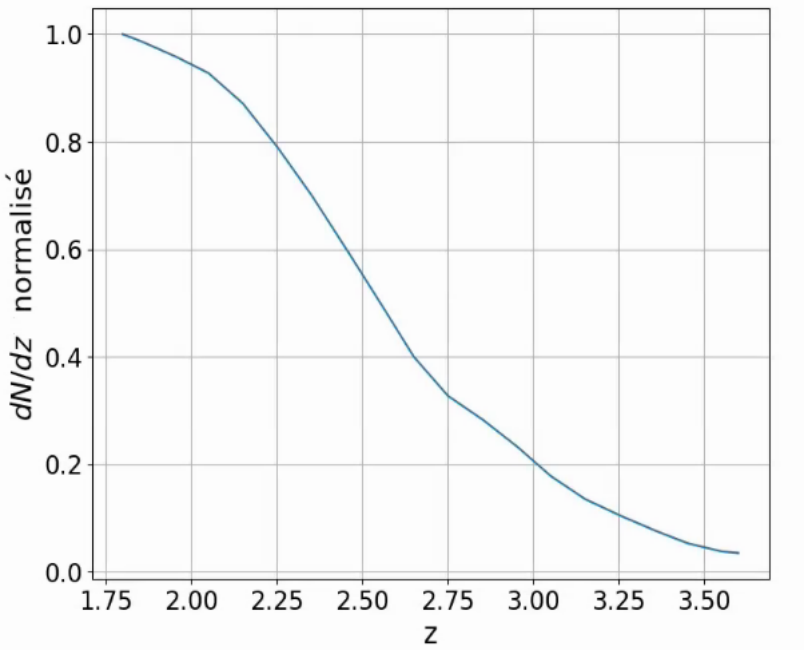

FIGURE 1.2 – Distribution normalisée en redshift des quasars tirés dans les mocks.

Enfin, grâce au facteur de normalisation $N(z)$, les quasars sont tirés selon la distribution en z normalisée prédite pour DESI. Cette distribution est présentée sur la figure ??. Cependant, nous dirons environ deux fois plus de quasars, afin de pouvoir simuler, entre autre, la sélection des cibles à l'aide du code `quickquasars` (présenté dans la section ??). A la fin, nous obtenons environ 100 quasars à $z > 2{,}1$ par degré carré.

Une fois les quasars tirés, nous les déplaçons proportionnellement à leur vitesse $v_\parallel$ le long de la ligne de visée. Celle ci est définie comme

$$v_\parallel = \frac{v_{\rm X}{\rm X} + v_{\rm Y}{\rm Y} + v_{\rm Z}{\rm Z}}{\sqrt{{\rm X}^2 + {\rm Y}^2 + {\rm Z}^2}} \,. \tag{1.19}$$

Ainsi, un quasar situé à une distance R sera replacé le long de la ligne de visé à une distance

$$R \to R + \frac{1}{H(z)} v_\parallel(z) \,, \tag{1.20}$$

avec

$$v_\parallel(z) = (1+z)\frac{dG}{dz}\frac{H(z)}{H_0 \frac{dG}{dz}(z=0)} v_\parallel \,. \tag{1.21}$$

Le facteur $(1+z)$ vient de la conversion des distances en distances comobiles. Une fois tous les quasars déplacés, leur redshift est recalculé, puis ils sont stockés dans un catalogue. Pour chaque quasar, le catalogue contient leur position dans le ciel (α, δ), leur redshift avec et sans RSD, ainsi qu'un identifiant unique.

2.3 Création des lignes de visée

A cette étape, nous disposons d'un catalogue de quasars, corrélé avec le champ de densité $\delta_{matière}$ qui sera utilisé pour construire l'absorption Lyα. Nous pouvons donc créer les lignes de visées à partir de chaque quasar, et interpoler le champ de densité le long de celles-ci. Dans un premier temps, nous commençons par créer le vecteur en longueurs d'onde observées, sur lequel sera interpolé le champ de

densité. Nous choisissons une taille de pixel $d_{pix} = 0{,}2\,h^{-1}$ Mpc. Les limites $1{,}8 < z < 3{,}6$ en redshift se traduisent par des limites $3403{,}876 < \lambda < 5592{,}082\,$Å sur la longueur d'onde observée pour le Lyα. Nous ajoutons la limite basse des spectrographes de DESI : $\lambda_{min} = 3530\,$Å, que nous réduisons afin d'inclure certains métaux dans les forêts (voir **??**). Les longueurs d'onde observées couvrent donc $3476{,}1877 < \lambda < 5591{,}566\,$Å à l'aide de 6524 pixels.

Une fois ce vecteur en longueur d'onde obtenu, nous le positionnons dans les boîtes afin de construire la ligne de visée à partir de chaque quasar. D'abord, le pixel $\lambda_{\rm QSO} = (1 + z_{\rm QSO})\lambda_{{\rm Ly}\alpha}$ est placé à la position $({\rm X_{QSO}}, {\rm Y_{QSO}}, {\rm Z_{QSO}})$ du quasar, et le vecteur est dirigé vers l'observateur. Puis, pour chaque pixel i entre $\lambda_{min} = 3476{,}1877$ et $\lambda_{\rm QSO}$, la position $({\rm X}_i, {\rm Y}_i, {\rm Z}_i)$ du pixel est déterminée. Le champ est alors interpolé puis lissé. Pour chaque pixel, nous considérons les voxcells appartenant au cube de 7 voxcells de côté, centré sur le voxcell dans lequel se trouve le pixel. Ceci représente donc 343 pixel à interpoler puis lisser. Le pixel i est alors donné par

$$\delta_i = \sum_{j=0}^{342} \frac{\delta(\vec{r}_j) {\rm e}^{\frac{-(\vec{r}_j - \vec{r}_i)^2}{\sigma_j^2}}}{\sigma_j^2} \,, \tag{1.22}$$

#prov c'est pas correcte (voir le code)

où $\vec{r}_j$ est la position du voxcell j, $\vec{r}_i$ celle du pixel i, δ est le champ à interpoler, et enfin $\sigma^2 = 2d_{cell}^2$ est la largeur du lissage gaussien appliqué. Ce lissage est nécessaire afin d'éviter le crénelage aux petites échelles. Ce calcul est effectué pour tous les pixels qui vérifient $\lambda_i < \lambda_{\rm QSO}$, pour chaque quasar. Les champs interpolés sont le champ $\delta_{matière}$ utilisé pour construire l'absorption Lyα, les trois champs de vitesse utilisés pour ajouter les RSD aux HCD tirés dans chaque ligne de visée (voir section 2.5), et les six champs de gradient de vitesse afin d'ajouter les RSD au champ Lyα.

2.4 De la densité à l'absorption

Une fois les lignes de visés interpolées, nous pouvons transformer le champ de densité en absorption Lyα. Ceci est fait via la formule FGPA :

$$F = \exp\Big(-a(z){\rm e}^{b(z)G(z)\delta_{matière}}\Big) \,. \tag{1.23}$$

Les petites échelles

Le champ $\delta_{matière}$, utilisé dans l'équation 1.23, est le champ de matière à grand échelle, construit grâce à la transformation de Fourier du champ δ_k. Cependant, ce champ est construit sur une grille de taille $d_{cell} = 2{,}19\,h^{-1}$ Mpc. Par conséquent, la plus petite échelle accessible est

$$k_N = \frac{2\pi}{d_{cell}} \sim 2{,}87\,h \text{ Mpc}^{-1} \,. \tag{1.24}$$

Nous manquons donc toutes les fluctuations pour lesquelles $k > k_N$. Sans ces fluctuations le spectre de puissance à une dimension, défini comme

$$P^{1D}(k_\parallel) = \frac{1}{2\pi}\int_{k_\parallel}^{\infty} kP(k)dk \,, \tag{1.25}$$

ne possède pas la bonne amplitude. De plus, ce sont ces fluctuations aux petites échelles qui contribuent principalement à la variance du champ. Le champ F construit ne possède donc pas le bon niveau de bruit. Pour palier ce problème, nous rajoutons indépendamment sur chaque ligne de visée un champ δ_s (*small scales* : petites échelles) au champ $\delta_{matière}$ que nous appelons désormais δ_l (*large scales* : grandes échelles) :

$$F = \exp\Big(-a(z){\rm e}^{b(z)G(z)(\delta_l + \delta_s)}\Big);. \tag{1.26}$$

Du fait que ce champ ne soit pas corrélé d'une ligne de visée à une autre, il ne participe pas à la fonction de corrélation à trois dimensions [1]. Afin d'ajouter la bonne quantité de fluctuations aux petites échelles, pour chaque ligne de visée nous générons un GRF à une dimension $\delta_{k,s}$, de la taille de la forêt. Puis, nous multiplions $\delta_{k,s}$ par

$$\sqrt{\frac{P_{miss}(k)}{d_{pix}}}\,, \tag{1.27}$$

où P_{miss} est le spectre de puissance qu'il faut appliquer à $\delta_{k,s}$ afin que F possède le bon P^{1D}. Le calcul de P_{miss} est détaillé dans la section **??**. Enfin, nous obtenons δ_s à l'aide de la transformation de Fourier de $\delta_{k,s}$.

Les RSD

Une fois les petites échelles ajoutées, nous obtenons un champ d'absorption F qui possède le bon spectre de puissance à 3 dimension pour les échelles $k_N < k < k_{max}$, avec

$$k_{max} = \frac{2\pi}{1536 d_{cell}} \sim 1{,}9 \times 10^{-3}\, h \text{ Mpc}^{-1}\,, \tag{1.28}$$

ainsi que le bon spectre de puissance à une dimension. Cependant, le champ d'absorption F ne possède pas de RSD pour l'instant, car il est construit à partir du spectre de puissance $P_{matière}(k)$ qui est isotrope. Initiallement, nous pensions ajouster les RSD au niveau du spectre de puissance : multiplier le GRF intial par $(1+\beta\mu^2)P_{matière}(k,\mu)$, avec $\mu = k_z/k$. Cependant, du fait que les lignes de visées ne sont pas parrallèles (elles l'étaient pour les mocks précédents, développés pour BOSS), nous ne pouvons pas confondre l'axe k_z avec l'axe de la ligne de visée $k_\parallel$. Nous avons alors choisi d'utilisé le champ de gradient de vitesse $\eta_\parallel$, présenté dans la section 2.1. Plusieurs essais (#prov les détailler ?) ont été menés afin de savoir comment inclure correctement le champ $\eta_\parallel$ dans FGPA. Nous présentons dans les lignes qui suivent la solution retenue. Le champ $\eta_\parallel$ est ajouté, en plus du champ δ_s, au champ δ_l. Ceci nous permet de retrouver la formule de Kaiser (équation 1.11). Afin de gérer la quantité de RSD, nous ajoutons un coefficient c, qui dépend de z. L'ajustement de ce paramètre nous permet d'obtenir la bonne dépendance en z pour $\beta_{\text{Ly}\alpha}$. La formule FGPA devient donc

$$F = \exp\Big(-a(z)\mathrm{e}^{b(z)G(z)(\delta_l+\delta_s+c(z)\eta_\parallel)}\Big);. \tag{1.29}$$

Les champs δ_l, δ_s et $\eta_\parallel$ sont calculés à $z = 0$, la dépendance en z étant prise en compte par le facteur $G(z)$. De plus, le facteur f que nous avons laissé de cotê dans la section 2.1 n'est pas explicité ici : pour les redshifts $z > 2$, l'univers est dominé par la matière et donc, en bonne approximation, nous avons $f(z) \sim 1$. Les paramètres $a(z)$, $b(z)$, $c(z)$, ainsi que $P_{miss}(z)$ sont ajustés afin d'obtenir le bon $b_{\text{Ly}\alpha}(z)$, $\beta_{\text{Ly}\alpha}(z)$, $\overline{F}(z)$ et $P^{1D}_{\text{Ly}\alpha}(z)$. L'ajustement est décrit dans la section **??**.

La prédiction

Il aurait été possible d'implémenter les RSD différemment. Une solution serait, par exemple, de déplacer chaque pixel d'absorption proportionnellement à la vitesse particulière du gaz dans la cellule considéré, puis de modifier l'absorption en fonction de gradient de vitesse dans cette cellule. En effet, si le gradient de vitesse est non nul, le gaz se retrouve comprimé par endroit, et détendu dans d'autres, modifiant l'absorption dans chaque cellule. Cette méthode pour ajouter les RSD dans des mocks Lyα est la méthode choisie par **?**. La méthode que nous utilisons, décrite dans la section précédente, a

1. Lors du calcul de la fonction de corrélation à 3 dimensions, nous ne considérons pas les paires de pixels issues de la même forêt (voir **??**)

l'avantage d'avoir une fonction de corrélation prédictible. C'est pour cela que nous avons fait le choix de cette méthode. Dans les lignes qui suivent, nous expliquons comment calculer la prédiction de la fonction de corrélation. Comme décrit par **?**, il est possible de relier la fonction de correlation ξ_F du champ F à la fonction de corrélation ξ_g du champ δ_g. Ces deux fonctions de corrélations sont reliées par

$$\xi_F(r_{12}) = \int_{-\infty}^{\infty} d\delta_{g1} \int_{-\infty}^{\infty} d\delta_{g2} \frac{\exp\left[-\frac{\delta_{g1}^2+\delta_{g2}^2-2\delta_{g1}\delta_{g2}\xi_g(r_{12})}{2(1-\xi_g^2(r_{12}))}\right]}{2\pi\sqrt{1-\xi_g^2(r_{12})}} \delta_F(\delta_{g1})\delta_F(\delta_{g2}) \,, \tag{1.30}$$

où δ_g est un GRF de variance 1, δ_F est le champ d'absorption calculé à partir du champ gaussien, et r_{12} est la distance qui sépare les deux points où sont évalués les champs δ_g et δ_F. Dans notre cas, le champ δ_g représente le champ $G(z)(\delta_l + \delta_c + c(z)\eta_\parallel)$. Ce champ est un champ gaussien, de valeur moyenne nulle et de variance σ_g^2. Nous compensons le fait que $\sigma_g^2 \neq 1$ en remplaçant ξ_g par ξ_g/σ_g^2 dans l'équation précédente. L'équation 1.30 ne dépendant que de la valeur de ξ_g, nous construisons une table qui permet de relier chaque valeur de $\xi_g \in [-1;1]$ à la valeur ξ_F correspondante. De plus, nous pouvons relier la fonction de corrélation dans l'espace des redshifts $\xi_g(r,\mu)$, à la fonction de corrélation $\xi(r)$ que suit le champ δ_l :

$$\xi(r,\mu) = \xi_0(r) + \frac{1}{2}\left(3\mu^2-1\right)\xi_2(r) + \frac{1}{8}\left(35\mu^4-30\mu^2+3\right)\xi_4(r) \,, \tag{1.31}$$

avec

$$\xi_0(r) = \left(1+\frac{2}{3}f+\frac{1}{5}f^2\right)\xi(r) \,, \tag{1.32}$$

$$\xi_2(r) = \left(\frac{4}{3}f+\frac{4}{7}f^2\right)\left[\xi(r)-\overline{\xi}(r)\right] \,, \tag{1.33}$$

$$\xi_4(r) = \frac{8}{35}f^2\left[\xi(r)+\frac{5}{2}\overline{\xi}(r)-\frac{7}{2}\overline{\overline{\xi}}(r)\right] \,, \tag{1.34}$$

et

$$\overline{\xi}(r) = 3r^{-3}\int_0^r \xi(s)s^2 ds \,, \tag{1.35}$$

$$\overline{\overline{\xi}}(r) = 5r^{-5}\int_0^r \xi(s)s^4 ds \,. \tag{1.36}$$

Ces équations sont décrites dans **?**. Afin d'obtenir la prédiction $\xi_F^{pred}(r,\mu)$, nous commençons par calculer le spectre de puissance que suit le champ δ_l :

$$P(k) = W^2(k)P_{matière}(z=0) \,, \tag{1.37}$$

où W est le terme représentant le lissage gaussien appliqué au champ δ_l (voir section 2.3) :

$$W(k) = \mathrm{e}^{-\frac{k^2 d_{cell}^2}{2}} \,. \tag{1.38}$$

(#prov il y a pas un facteur $\frac{1}{2}$ en trop ? on a $\sigma_{smooth}^2 = 2d_{cell}^2$)

A l'aide d'une transformation de Fourier, nous obtenons la fonction de corrélation $\xi(r)$ que suit le champ interpolé δ_l. Puis, nous calculons la fonction de corrélation dans l'espace des redshifts $\xi_g(r,\mu)$ que suit le champ δ_g gràce à l'équation 1.31. Enfin, pour tous les couples (r,μ) nécessaires, nous obtenons la fonction de corrélation $\xi_F(r,\mu)$ du champ F comme la valeur correspondante à la valeur tabulée $\xi_g(r,\mu)/\sigma_g^2$ pour ξ_g. (#prov un plot de la prédiction ?)

2.5 Ajout des HCD

Les HCD ont un effet important dans les fonctions de corrélation, nous simulons donc aussi leur présence. De manière à avoir une corrélation entre les HCD et les autres traceurs des mocks, nous utilisons le champ de densité δ_l pour tirer les HCD. Nous ne considérons pas la somme $\delta_l + \delta_s$ car les HCD sont des surdensités à grandes échelles : une résolution de 2,19 h^{-1} Mpc est suffisante. De plus, l'ajout de δ_s bruiterait la corrélation entre les HCD et les autres traceurs.

Contrairement aux quasars, les HCD sont tirés proportionnellement au champ δ_l. Nous identifions les cellules dans lesquelles δ_l est au dessus d'un certain seuil, puis les HCD sont tirés dans ces cellules selon une loi de Poisson. Le seuil ν est défini en fonction du biais souhaité pour les HCD. Pour un seuil ν, le biais obtenu est donné par

$$b_\nu = \frac{pdf(\nu)}{cdf(-\nu)} \,, \tag{1.39}$$

où $pdf(\nu)$ donne la densité de probabilité de ν, et $cdf(-\nu)$ est la fonction de répartition : la probabilité d'être au dessus du seuil ν. Ainsi, pour avoir un biais de 2, il faut que la probabilité que le champ prenne la valeur ν soit 2 fois plus grande que la probabilité que le champ soit au dessus du seuil ν. Afin d'obtenir le seuil pour un biais donné, nous calculons b_ν pour une large gamme de seuils ν puis nous interpolons b_ν sur ν. Ainsi, pour un biais b, nous connaissons le seuil $\nu(b)$ à choisir. Dans notre cas, le champ δ_l suit une densité de probabilité gaussienne. Cependant, sa variance n'est pas égale à 1. De plus, le champ δ_l interpolé le long des lignes de visée correspond au champ de matière à $z = 0$. Ainsi, pour obtenir un biais $b_{\rm HCD}$, pour chaque redshift nous calculons le seuil ν comme si nous visions un biais $b = b_{\rm HCD}\sigma_l G(z)$. Le terme σ_l prend en compte la variance du champ δ_l, et $G(z)$ le fait que δ_l soit construit à $z = 0$. Une fois les cellules pouvant héberger un HCD identifiées, nous tirons dans chacune d'entre elles les HCD avec une loi de poisson de paramètre

$$\lambda(z) = \frac{N(z)}{cdf(-\nu(z))} \,, \tag{1.40}$$

où $N(z)$ donne le nombre moyen de HCD attendu par cellule et $\nu(z)$ le seuil au redshift z. Le nombre de HCD attendu est donné par la librairie `pyigm` [1]. La distribution en redshift des HCD est présenté sur le graphique de gauche de la figure 1.3. Une fois tous les HCD tirés, nous leur assignons une densité de colonne dans la gamme $17{,}2 < \log(n_{HI}) < 22{,}5$, selon la distribution donnée par `pyigm`. Cette distribution est présentée sur le graphique de droite de la figure 1.3. Enfin, nous ajoutons les RSD aux HCD tirés. Chaque HCD tiré est déplacé le long de la ligne de visée proportionnellement à la vitesse

$$v_\parallel = \frac{v_{\rm X}{\rm X} + v_{\rm Y}{\rm Y} + v_{\rm Z}{\rm Z}}{\sqrt{{\rm X}^2 + {\rm Y}^2 + {\rm Z}^2}} \,, \tag{1.41}$$

où $v_{\rm X}$, $v_{\rm Y}$ et $v_{\rm Z}$ sont les champs de vitesse interpolés le long de la ligne de visée. Ainsi, un HCD à un redshift z sera déplacé à un redshift

$$z \to z + (1+z)H(z)\frac{dG}{dz}\frac{1}{H_0 \frac{dG}{dz}(z=0)} v_\parallel \,. \tag{1.42}$$

Dans les mocks que nous décrivons ici, le profil d'absorption des HCD n'est pas ajouté dans les forêts. Nous produisons uniquement un catalogue qui regroupe tous les HCD tirés. Le profile d'absorption est ajouté au spectre de chaque quasar par le code `quickquasars`, qui utilise le catalogue de HCD que nous produisons.

1. https ://github.com/pyigm

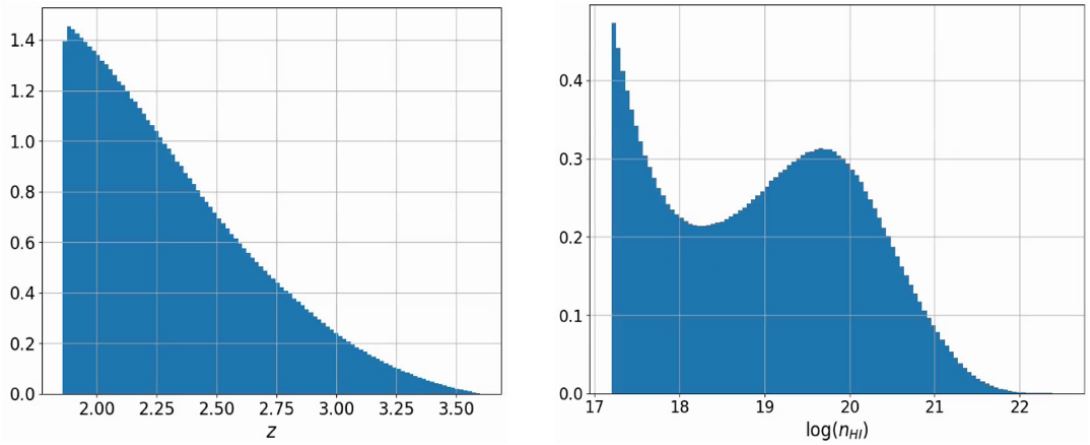

FIGURE 1.3 – Gauche : distribution normalisée en redshift des HCD. Droite : distribution normalisée de $\log(n_{HI})$ des HCD. Ces distributions proviennent de la librairie `pyigm`.

3 Production des mocks

Comme expliqué au début de ce chapitre, l'objectif des mocks est de reproduire les données d'eBOSS et de DESI. Etant donné que le relevé d'eBOSS est contenu dans le relevé de DESI, nous simulons directement le relevé DESI. Ainsi, lorsque nous avons besoin de simuler le relevé d'eBOSS, nous retirons les quasars qui ne sont pas contenu dans ce relevé. La taille des boîtes choisie ($2560 \times 2560 \times 1536$ voxcells) et leur résolution ($2{,}19\,h^{-1}$ Mpc) ne suffisent pas à couvrir les 14 000 degrés carrés de DESI. Pour palier ce problème, nous construisons sept *chunks* indépendants, que nous assemblons pour former le relevé de DESI. Le découpage du relevé en sept chunks est montré sur la figure **??**. Ce choix d'assembler sept boîtes de densité plutôt que d'en utiliser une seule a été contraint par la mémoire maximale des noeuds. Les mocks ont été produit grâce au centre de calcul NERSC [1], avec la machine Cori. Sur cette machine, nous avons utilisé les noeuds "Haswell", au nombre de 2388. Chaque noeud possède 32 cœurs, chacun possèdant 2 hyper-threads. Chaque noeud peut donc gérer 64 tâches simultanément. De plus, ces noeuds possèdent 128 Go de mémoire vive. De manière à créer nos boîtes de densité, via la transformation de Fourier à trois dimensions, il faut, pour chaque boîte, que l'intégralité de son contenu soit accessible depuis un même endroit. Nous pourrions distribuer la mémoire et effectuer la transformation de Fourier sur plusieurs noeuds à l'aide de la librairie MPI. Cependant nous avons choisi de ne pas utiliser cette librairie et d'effectuer les transformations de Fourier sur un seul noeud, ce qui explique le découpage du relevé en sept chunks indépendants. Nous profitons néanmoins des 64 threads de chaque noeud pour paralléliser notre code.

Dans les lignes qui suivent, nous détaillons les différents éléments du code. Le schéma **??** résume la situation. Le première module, `interpolate_pk.py` permet de calculer puis d'interpoler les quatres spectres de puissance $P_{\mathrm{QSO},i}$ et $P_{matière}$ sur la grille $2560 \times 2560 \times 1536$. Le code est lancé séparément sur 16 morceaux de la boîte, puis le code `merge_pk.py` permet de rassembler des 16 morceaux des spectres de puissance interpolés et de les sauver au format FITS (Flexible Image Transport System). Ce étape est effectuée une seule fois, car les spectres de puissance sont communs à toutes les réalisations. Le module suivant est `make_boxes.py`. Ce code lit les spectres de puissance interpolés et construit les différents champs de densité, de vitesse et de gradient de vitesse décrits dans la section 2.1. Ce code est lancé sept fois, afin de produire les champs pour les septs chunks. Une fois tous les champs produits, les quasars sont tirés (section 2.2) grâce au code `draw_qso.py`. Afin d'accélerer la production des mocks,

1. National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility operated under Contract No. DE-AC02-05CH11231.

les boîtes sont partagés en 512 tranches selon l'axe x. Ces tranches, de taille $5 \times 2560 \times 1536$, sont traitées séparément. Ce code est donc tourné en parallèle 512 fois, sur 16 noeuds $\times$ 32 threads. Une fois les quasars tirés, les lignes de visée sont interpolées (section **??**) avec le code `make_spectra.py`. De la même manière que le code précédent, il tourne en parallèle 512 fois, sur 16 noeuds $\times$ 32 threads. Une fois tous les quasars traités, les morceaux de ligne de visée sont mis bout à bout grâce au code `merge_spectra.py`. Encore une fois, le code tourne en parallèle 512 fois. Lorsque les lignes de visée sont reconstruites, la formule FGPA est appliquée afin d'obtenir le champ de transmission F pour chaque ligne de visée. La dernière étape consiste à regrouper le résultat de chaque chunk. Le module `merge_qso.py` permet de lire tous les quasars tirés dans chaque tranche de chaque chunk et de créer un catalogue global appelé `master.fits`. Une fois le catalogue construit, le code `dla_saclay.py` tire les HCD dans chaque ligne de visée. Le code est tourné sur les septs chunks en parallèle. Puis le module `merge_dla.py`, comme pour les quasars, permet de regrouper tous les HCD tirés et de construire le catalogue global `master_DLA.fits`. Enfin le code `make_transmissions.py` permet de mettre les fichiers contenant les forêts au bon format : les forêts sont regroupées par HEALPix pixel (#prov expliquer ou mettre une ref ?) et sont stockées dans les fichiers

`N/PIX/transmission-nside-PIX.fits.gz` ,

où `nside` = 16 est la résolution du schéma HEALPix utilisé, `PIX` donne le numéro du pixel HEALPix, et `N` est le résultat de la division euclidienne de `PIX` par 100. Cette dernière étape est effectuée sur un noeud, `make_transmissions.py` étant lancé en parallèle sur 64 sous-échantillons des HEALPix pixels.

Pour les analyses Lyα d'eBOSS et de DESI, nous avons décidé de produire 100 réalisations des mocks, afin d'avoir suffisamment de statistique pour étudier finement les potentielles systématiques. Nous avons organisé la production de ces 100 réalisations en deux étapes. Premièrement, nous avons effectué la *pré-production*. Cette étape consiste à créer les boîtes contenant les différents GRF, puis à tirer les quasars, reconstruire la densité, les vitesses et gradients de vitesse, le long de chaque ligne de visée. Ceci correspond aux codes `make_boxes.py`, `draw_qso.py` et `make_spectra.ppy`. Cette étape est la plus coûteuse en temps de calcul : environ 43 heures CPU sur un noeud de Cori pour produire les septs chunks. L'ensemble des fichiers temporaires propres à une réalisation pré-produite est estimé à $\sim 7 \times 550\,$Go sur disque. Cependant, nous stockons uniquement les lignes de visée interpolées, et la boîte δ_k, ce qui représente $\sim 340\,$Go sur disque, par réalisation.

Une certain nombre de problèmes informatiques, liés au centre de calcul NERSC, ont ralenti la phase de pré-production. Le principal problème venait du temps de lecture et d'écriture, qui par moment pouvait être multiplié par un facteur cent. Plus de trentes secondes étaient parfois nécessaires pour accéder à un simple fichier. Ce problème a été identifié comme venant du transfert des *meta data* sur le centre de calcul. Le code le plus affecté est `make_boxes.py`, car il écrit énormément de fichiers différents, correspondant aux différents slices des boîtes, et destinés à être lus par les codes `draw_qso.py` et `make_spectra.py`. Pour palier ce problème, nous avons décidé d'essayer de faire tourner les codes sur les noeuds de Cori appelés *Burst Buffer*. Ces noeuds possèdent des disques SSD (Solid-State drive), ce qui permet une lecture et une écriture très rapide. Une fois les codes exécutés, les données produites sont déplacées sur les disques durs habituels. Les noeuds Burst Buffer ont permis d'accélérer l'exécution du code `make_boxes.py` par un facteur ~ 2, passant d'environ deux heures à une heure seulement. Une vintaine de réalisation ont été produites en utilisant ces noeuds. Cependant, ils sont devenus instables au cours de la production. D'autre part, les problèmes liés aux transferts des meta data avaient été stabilisés entre temps. Nous sommes donc retournés à l'utilisation des noeuds classiques de Cori pour finir la production. Pour les 40 dernières réalisations, le temps d'exécution de `make_boxes.py` variait entre trois et quatres heures.

Le temps laissé par les six mois nécessaires à la production nous a permis de choisir les paramètres Lyα souhaités, et d'ajuster les paramètres de la formule FGPA (équation 1.29) en conséquence (voir

section **??**). Une fois ces paramètres ajustés et la phase de pré-production terminée, nous avons mené la phase de *post-production*. Cette phase consiste à créer les spectres d'absorption à partir des densités interpolées le long des lignes de visée, puis à regrouper tous les fichiers de sortie afin de les mettre au format décrit précédemment. Cette étape est beaucoup plus rapide, elle prend l'équivalent d'environ 6 heures CPU sur un noeud de Cori. Une fois la production complète effectuée, la place sur disque d'une réalisation, sans compter les fichiers temporaires, correspond à environ 15 Go sur disque.

Bibliographie

Aniket Agrawal, Ryu Makiya, Chi-Ting Chiang, Donghui Jeong, Shun Saito, and Eiichiro Komatsu. Generating {Log}-normal {Mock} {Catalog} of {Galaxies} in {Redshift} {Space}. *arXiv :1706.09195 [astro-ph]*, jun 2017. URL http://arxiv.org/abs/1706.09195.

L. Clerkin, D. Kirk, M. Manera, O. Lahav, F. Abdalla, A. Amara, D. Bacon, C. Chang, E. Gaztañaga, A. Hawken, B. Jain, B. Joachimi, V. Vikram, T. Abbott, S. Allam, R. Armstrong, A Benoit-Lévy, G. M. Bernstein, E. Bertin, D. Brooks, D. L. Burk, A. Carnero Rosell, M. Carrasco Kind, M. Crocce, C. E. Cunha, C. B. D'Andrea, L. N. da Costa, S. Desai, H. T. Diehl, J. P. Dietrich, T. F. Eifler, A. E. Evrard, B. Flaugher, P. Fosalba, J. Frieman, D. W. Gerdes, D. Gruen, R. A. Gruendl, G. Gutierrez, K. Honscheid, D. J. James, S. Kent, K. Kuehn, N. Kuropatkin, M. Lima, P. Melchior, R. Miquel, B. Nord, A. A. Plazas, A. K. Romer, E. Sanchez, M. Schubnell, I. Sevilla-Noarbe, R. C. Smith, M. Soares Santos, F. Sobreira, E. Suchyta, M. E. C. Swanson, G. Tarle, and A. R. Walker. Testing the lognormality of the galaxy and weak lensing convergence distributions from Dark Energy Survey maps. may 2016. doi : 10.1093/mnras/stw2106. URL http://arxiv.org/abs/1605.02036http://dx.doi.org/10.1093/mnras/stw2106.

Antony Lewis, Anthony Challinor, and Anthony Lasenby. Efficient Computation of CMB anisotropies in closed FRW models. nov 1999. doi : 10.1086/309179. URL http://arxiv.org/abs/astro-ph/9911177http://dx.doi.org/10.1086/309179.