Command-line interfaces and tools for biologists

Pierre Tocquin (ptocquin@uliege.be)

2021

Contents

List of Boxes

List of Tables

List of Figures

Chapter 1

Linux: Installation et découverte du système d'exploitation.

1.1 Pourquoi Linux ?

1.1.1 Le système d'exploitation du domaine *Data Science*.

Si on se réfère aux statistiques de popularité des systèmes d'exploitation, on peut légitimement se poser la question de l'intérêt de consacrer du temps et de l'énergie à apprendre à maitriser un système d'exploitation tel que Linux qui n'équipe que 2% des ordinateurs de bureaux en Europe (Figure 1.1).

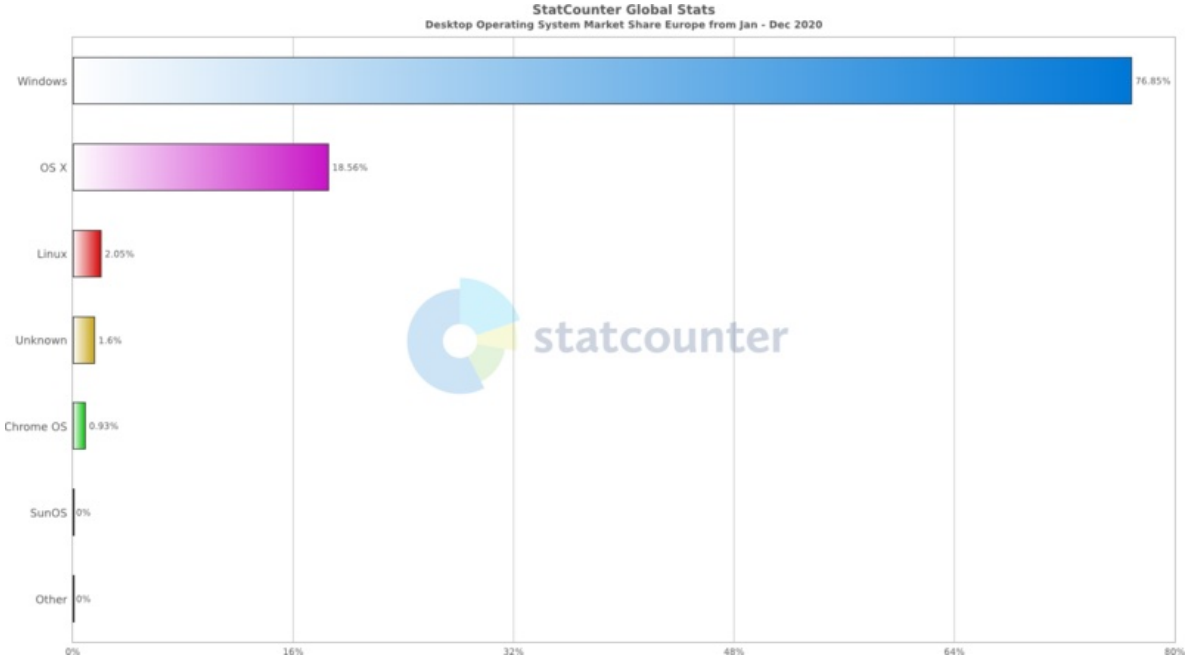

Figure 1.1: Statistiques d'usage des systèmes d'exploitation en Europe en 2020. Source: StatCounter.com

Ces chiffres sont évidemment le résultat d'une analyse menée pour un usage "grand public" (il s'agit

de données récoltées lors de la consultation de sites internet). Or, lorsqu'on s'éloigne d'une utilisation domestique d'un ordinateur, l'usage spécifique qui en est fait dicte généralement de manière importante le choix du système d'exploitation. Ainsi, il est remarquable que, chez les "gamers", Microsoft Windows soit quasiment l'unique système utilisé (96,5% en juin 2021. Source: Steam). A l'inverse, si on s'intéresse aux systèmes qui équipent les super-calculateurs utilisés pour le traitement de données, on constate que **la totalité** des 500 plus grosses machines au niveau mondial tournent sous Linux (Source: top500.org).

Comme nous ne sommes pas là pour jouer mais bien pour apprendre les bases de l'analyse de données biologiques, nous pourrions en rester là dans l'argumentation en faveur de Linux… Mais prenons néanmoins le temps d'évoquer l'origine de ce système d'exploitation et de lister quelques éléments concrêts qui expliquent cette égémonie de Linux dans le domaine du *Data Science*.

1.1.2 UNIX-based et *open-source*.

Au delà du monopole de Linux sur les machines de calcul, ce système d'exploitation est certainement le plus populaire chez les développeurs informatiques, en particulier dans le monde académique qui est à l'origine de Linux.

L'histoire de Linux est intimement liée à celle d'un autre système d'exploitation nommé UNIX. UNIX est né à la fin des années '60 en parallèle avec le langage C. Ses concepteurs, Ken Thompson et Dennis Ritchie, ont le souhait de produire un système d'exploitation **multi-utilisateurs** et **multi-tâches** pouvant être installé sans trop de difficultés sur différents modèles de machines. Ces 2 développeurs travaillent à l'époque pour le Bell Labs de la société de télécommunication AT&T (aujourd'hui Nokia Bell Labs). A l'époque, pour des raisons légales, AT&T ne peut pas commercialiser UNIX et décide donc de le diffuser notamment vers les universités avec une licence peu contraignante: cela va doper le développement de UNIX. À cette époque, l'Université de Californie à Berkeley est l'un des plus important contributeur au développement d'une version open-source de UNIX (*Berkeley Software Distribution* ou BSD).

Notez qu'un grand nombre des systèmes d'exploitations utilisés aujourd'hui, dont MacOS mais à l'exception notable de Microsoft Windows, sont issus ou inspirés de UNIX. La généalogie complète (et impressionante!) de UNIX peut être visualisée sur `https://www.levenez.com/unix/`. La multiplication des systèmes UNIX-like a créé le besoin du développement d'une norme permettant la standardisation des interfaces de programmation et des outils qui composent un système UNIX. Cette norme, dénomée POSIX (Portable Operating System Interface, le 'X' rappelant UNIX), a été créée en 1988 et a subi, depuis, une série d'évolution, la dernière datant de 2017.

Jusque dans les années '70, le développement des ordinateurs et de l'informatique est l'apanage des universités. Les idées et les codes informatiques circulent et sont échangés librement. Cette situation évolue négativement, peu à peu les programmes informatiques sont vus comme des produits comemrcialisables que l'on protège par des licences plus restrictives et prohibitives. UNIX n'échappe pas à cette tendance. En réaction, Richard Stallman décide de créer un nouveau système d'exploitation complètement libre qu'il appèlera GNU (**G**NU's **N**ot **U**NIX, qui se prononce "gnou"). Il en fait un système calqué sur UNIX et compatible avec lui. Il adopte donc la même phylosophie qui est d'utiliser de petits programmes qui chacun ne font qu'une seule chose (mais la font bien) et sont capables de collaborer entre eux (McIlroy et al., 1978). Il ne manque qu'une seule chose à GNU: un noyau, c'est à dire la partie du système d'exploitation qui gère le matériel. Ce manque sera comblé en 1991 lorsqu'un

étudiant finlandais entreprendra de développer un système d'exploitation pour utiliser son ordinateur à base de processeur Intel. Il développera un nouveau noyau, Linux, et y greffera les outils GNU: GNU/Linux est né.

En tant que tel, Linux n'est donc pas un système d'exploitation complet. Il s'agit seulement du *noyau* contenant l'ensemble des programmes permettant de gérer le matériel: gestion de la mémoire, accès au disques, aux péripériques, etc… Il est donc plus exact de considérer le système d'exploitation comme la combinaison de Linux et de GNU. C'est pourquoi on parle de "GNU/Linux".

1.1.3 Le terrain de jeu de prédilection des bioinformaticiens

Linux supporte la majorité des langages de programmation, ce qui en fait un environnement de développement idéal. C'est l'une des raisons pour lesquelles l'essentiel des programmes de bioinformatique fonctionnent sous Linux (parfois exclusivement) et sont distribués avec une licence open-source.

Un autre élément important est que la ligne de commande Linux (ou Shell), est largement supérieure à celle de Windows. Nous le découvrirons à travers ce cours dont c'est l'objectif principal. Cela n'a d'ailleurs pas échappé à Microsoft qui a décidé d'intégrer à Windows 10 un environnement GNU/Linux, le *Windows Subsystem for Linux* (WSL) qui permet "… d'exécuter des outils en ligne de commande Linux natifs directement sur Windows, en même temps que votre bureau et vos applications Windows traditionnels" (Voir aussi Barnes (2021)).

Enfin le support natif du protocole de communication *ssh* rend très simple le travail en mode client/serveur qui est la règle: votre ordinateur de bureau ne doit généralement servir que de porte d'accès (de terminal) vers des machines puissantes permettant le traitement des tâches gourmandes en ressources, tant en stockage qu'en puissance de calcul. Il n'est pas rare que des analyses génomiques prennent déjà des heures, des jours ou des semaines sur ces machines ! Inutile de penser les réaliser sur votre portable.

1.2 Installation de la distribution Ubuntu

Le caractère open-source et modulaire de GNU/Linux a conduit à l'émergence d'une multitude de variantes, appelées distributions, chacune étant un ensemble cohérents de logiciels, le plus souvent eux-aussi open-source, fonctionnant autours du noyau Linux. Ces distributions peuvent se distinguer par leur ergonomie, leur interface graphique, les outils de gestion du système, leur cycle de mises à jour, etc… Certaines sont orientées "grand-public" et d'autres sont spécialisées pour des applications précises en entreprise.

Ubuntu est l'une des distributions GNU/Linux les plus populaires. Elle est née en 2005 au départ d'une autre distribution, Debian, avec pour objectif d'en faire une distribution simple à installer, conviviales et disposant d'un large catalogue de logiciels régulièrement mis à jour.

Cette distribution existe sous une forme Desktop ou Serveur (pas d'interface graphique). Elle est mise à jour de manière très régulière, deux fois par an en avril et en octobre. Tous les 2 ans, une version reçoit le label LTS (Long Term Support) qui dispose d'un noyau Linux récent et, comme son nom l'indique, est maintenue pendant une période prolongée de 5 ans. Ces versions LTS sont donc idéales lorsqu'on fait un usage professionnel de GNU/Linux.

1.2.1 Téléchargement de la dernière version LTS de Ubuntu et création d'une clé USB *bootable*

Un fichier ISO (image disque) est téléchargeable sur le site Ubuntu: `https://ubuntu.com/download/desktop`.

1.2.2 Essayer/Installer Ubuntu sur votre disque dur.

BOX 1: À vous de jouer > créez une clé USB d'installation

Munissez-vous ensuite d'une clé USB pouvant être complètement effacée et de minimum 4GB. Suivez ensuite les procédures décrites pour Windows et pour MacOS un peu plus bas sur la page de téléchargement ("Easy ways to switch to Ubuntu") pour pour créer un système Ubuntu *bootable* sur cette clé.

Configurer votre ordinateur pour qu'il utilise la clé USB comme support de démarrage. Cela se fait généralement en appuyant sur une touche lors des premières secondes après le démarrage de votre ordinateur (F12, ESC, F2, F10 ou une autre touche qui est généralement mentionnée brièvement à l'écran).

Lorsque le système démarre sur la clé USB, un menu apparait pour vous proposer soit d'installer, soit d'essayer Ubuntu. Cette dernière possibilité est intéressante, notamment si vous souhaitez tester quelques distributions avant de choisir celle qui vous convient. Dans ce cas, vous accéderez à une session Ubuntu identique à celle que vous auriez après installation sauf qu'aucune modification n'est apportée à votre système: tout se passe sur la clé USB.

Lorsque vous choisissez d'installer la distribution, vous serez guidé dans un processus d'installation simple et intuitif. Attention cependant, cette opération va nécessairement modifier la structure de votre disque dur. **Cela n'est pas sans risque et vous veillerez toujours à disposer d'une sauvegarde à jour des données présentes sur votre ordinateur**.

Un premier écran vous permet de configurer votre clavier. Ensuite, vous serez invités à choisir le type d'installation et d'autres options. Je vous suggère de choisir l'installation minimale et de décocher toutes les autres options.

L'écran suivant est le plus critique puisqu'il vous proposera soit d'effacer complètement votre disque pour y installer Ubuntu, soit d'installer Ubuntu "à côté" d'un système d'exploitation pré-existant. Notez que même dans ce dernier cas, vos données peuvent être endommagées ou perdues lors du processus ! Néanmoins, il se déroule généralement très bien et vous permet de choisir, lors du démarrage de votre ordinateur, le système que vous souhaitez utiliser. Dans cette configuration, appelée "dual-boot", il n'est donc pas possible de basculer d'un système à l'autre sans redémarrer votre machine. L'avantage est que chaque système peut exploiter 100% des ressources de votre ordinateur (à l'exception du disque dur qui est partagé lors du processus d'installation).

Avant que l'installation débute réellement, il reste à configuer le fuseau horaire, puis à introduire les informations de connexion pour l'utilisateur principal, ou administrateur, du système. Cette dernière opération est essentielle car l'utilisateur qui va être ainsi créé est celui qui disposera de tous les droits tels que ceux nécessaires à l'installation de nouveaux programmes, à la modification ou la suppression de fichiers systèmes, etc… Si vous perdez ces informations, il ne vous sera plus possible de modifier ou même d'accéder à votre système !

1.3 Découverte de l'environnement

1.3.1 L'interface graphique

L'interface graphique, ou Graphical User Interface GUI, fournie par défaut avec Ubuntu 20.04 est Gnome 3. Le caractère modulaire des distributions Linux implique qu'il existe une multitude d'environnement de bureau différents et il est donc toujours possible de ne pas conserver la solution fournie par défaut mais d'installer l'environnement graphique qui vous convient le mieux. Les GUI les plus connues sont Gnome, KDE et Mate (l'un des 3 GUI proposés par la distribution Linux Mint).

La découverte de cet environnement ne devrait pas vous poser de problèmes tant son ergonomie est grande et son organisation rappelle les autres systèmes d'exploitations, particulièrement MacOS. Des informations détaillées sur l'usage de ce GUI sont disponibles sur le site help.ubuntu.com.

BOX 2: À vous de jouer > configurez votre environnement

Découvrez l'outils de configuration 'Paramètres'. Pour y accéder, vous pouvez soit cliquer sur le bouton 'grille de points' en bas du *dash* (la barre verticale d'icônes), soit cliquer sur 'Activités' (en haut à gauche). Dans les 2 cas, vous pouvez rechercher rapidement les applications ou fichiers en tapant quelques lettres dans la barre de recherche. Parcourez les différents menus et configurer ce qui doit l'être, par exemple la connexion réseau, la langue de l'environnement, ou la disposition du clavier. Ce dernier élément est essentiel puisque très vite le clavier sera votre unique interface avec l'ordinateur.

1.3.2 Le système de fichiers

L'organisation du système de fichiers respecte les recommandations édictées pour les sytèmes UNIX par le *Filesystem Hierarchy Standard* (FHS). Il est essentiel d'en comprendre la topologie car la navigation dans ce système de fichiers se fera le plus souvent sans l'aide de l'interface graphique. A ce stade, profitez encore de l'interface graphique et de votre souris pour parcourir ce système de fichiers dans tous les sens via le navigateur de fichiers 'Files'.

Lorsque vous ouvrez cet outil, il affiche le contenu de votre répertoire utilisateur ou 'Home' (Figure 1.2).

Pour explorer l'ensemble du système de fichier, il est nécessaire de remonter à la *racine* de celui-ci. Pour cela, cliquez sur 'Autres emplacements' et choisissez 'Ordinateur': vous êtes alors à la base de l'arborescence et vous voyez le premier niveau de répertoires qui constituent votre système de fichiers (Figure 1.3).

Les éléments principaux de cette structure sont détaillés sur help.ubuntu.com/community/LinuxFilesystemTreeOverv
Consultez cette information et naviguez dans les différents répertoires pour en appréhender le contenu. Notez en particulier le répertoire *home*, celui-ci contient un autre répertoire dont le nom est celui de votre identifiant de connexion. Lorsque vous ouvrez ce répertoire, vous vous retrouvez au même endroit qu'au démarrage du navigateur de fichiers (Figure 1.2).

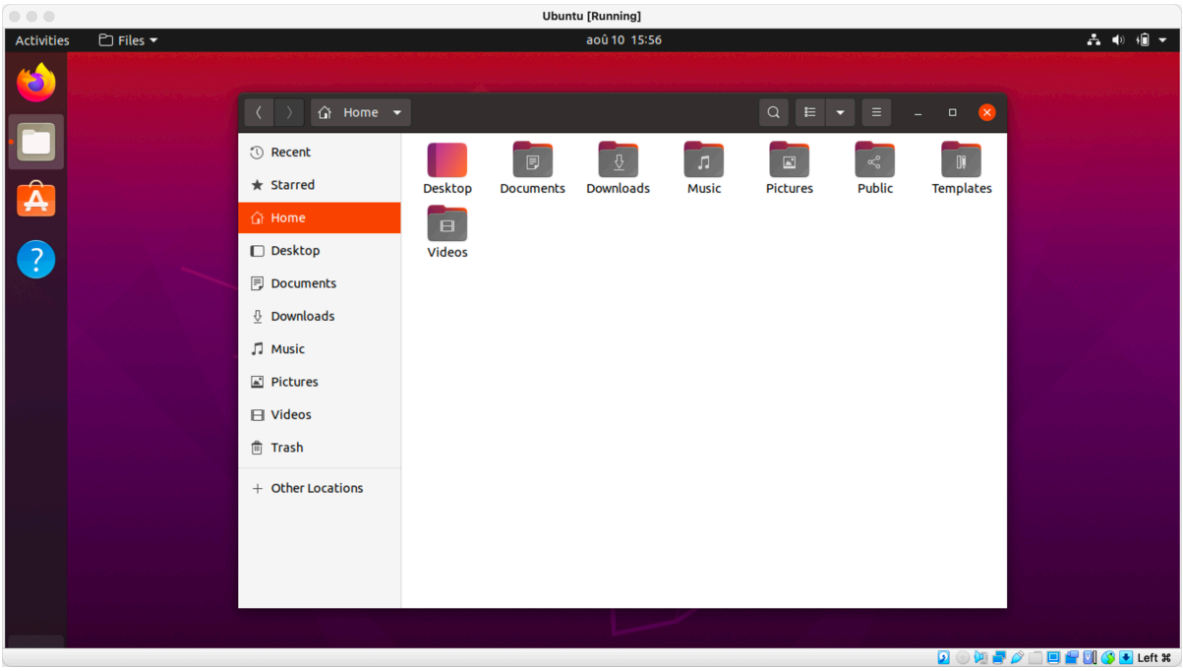

Figure 1.2: Exploration du système de fichiers à partir du répertoire utilisateur

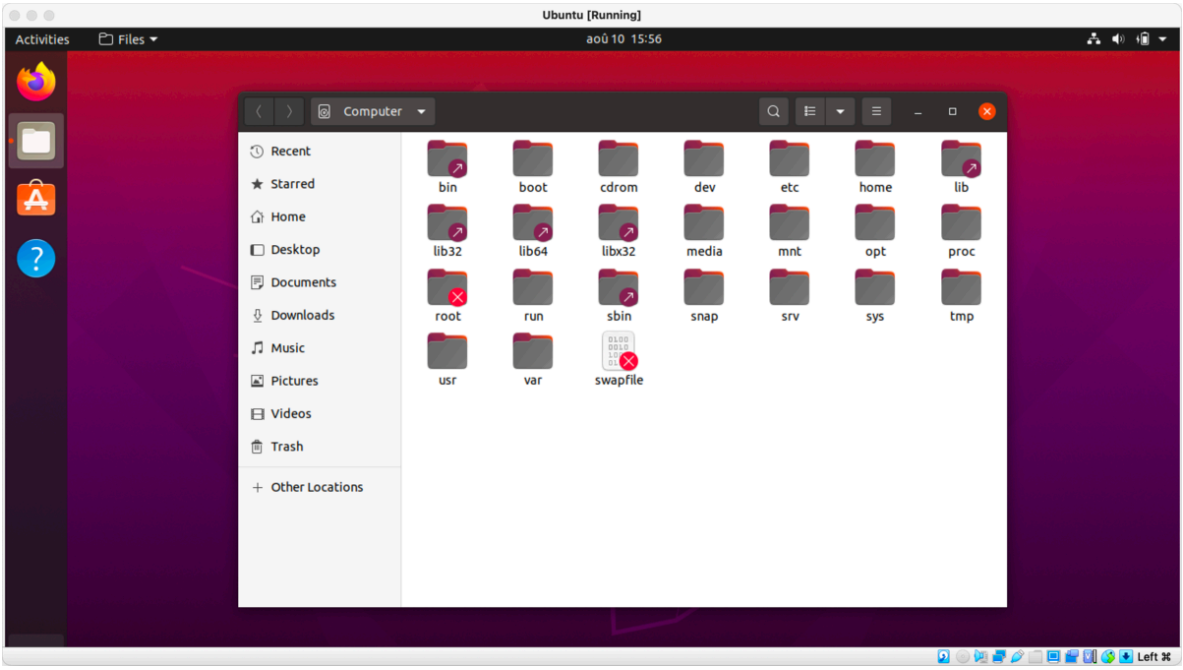

Figure 1.3: Exploration du système de fichiers à partir de la racine

Chapter 2

Le shell

Le shell est un logiciel au même titre que tous les autres programmes qui constituent votre système d'exploitation. Ce programme est un **interpréteur de commandes**. Il est la 'coquille' qui entoure le noyau linux et qui constitue donc l'interface entre l'utilisateur et ce noyau. Le shell

2.1 *Les* shells

Il existe de nombreux shell. Le premier, *sh*, a été développé en 1971 par Kenneth Thompson, l'un des créateurs de UNIX. Il fut remplacé quelques années plus tard par le Bourne shell (*sh*) qui devient la base de la plupart des shell développé par la suite, tels que C shell (*csh*) ou Korn shell (*ksh*). Un nouveau shell, Bourne-Again shell (*bash*) est développé en 1988 dans le cadre du projet GNU: il deviendra le shell par défaut de la plupart des systèmes UNIX et GNU/Linux, dont Ubuntu 20.04. Notez que le Z shell (*zsh*), développé en 1990, gagne en popularité car il reprend les fonctions de *bash* tout en l'enrichissant de fonctionnalités qui lui confère une plus grande ergonomie. Il est le shell par défaut de MacOS depuis la version Catalina.

2.2 Le terminal

Le shell a pour objet d'interpréter des commandes qui lui sont envoyées par l'utilisateur. Un autre programme, le terminal ou la console, fournit l'interface entre l'utilisateur (le clavier) et le shell. Lorsque le terminal s'ouvre, le shell est lancé automatiquement et est prêt à recevoir des commandes. Le curseur est placé à l'extrémité d'une ligne qui contient un certain nombre d'informations et qui se termine généralement par `$`. Cette ligne s'appelle l'invite de commande ou *prompt* (Figure 2.1).

2.3 Anatomie d'une commande

En mode interactif, l'invite de commande reçoit les caractères tapés au clavier. **Chaque caractère compte**, qu'il soit visible ou invisible. Ainsi, les espaces délimitent des *mots* qui, chacun, en fonction de leur ordre d'interprétation par le shell (de gauche à droite) ou des caractères (codes) qui le composent auront une fonction (une signification) différente. La casse (minuscules ou majuscules) est également interprétée (la ligne de commande est *case-sensitive*), ce qui aura notamment des conséquences sur le

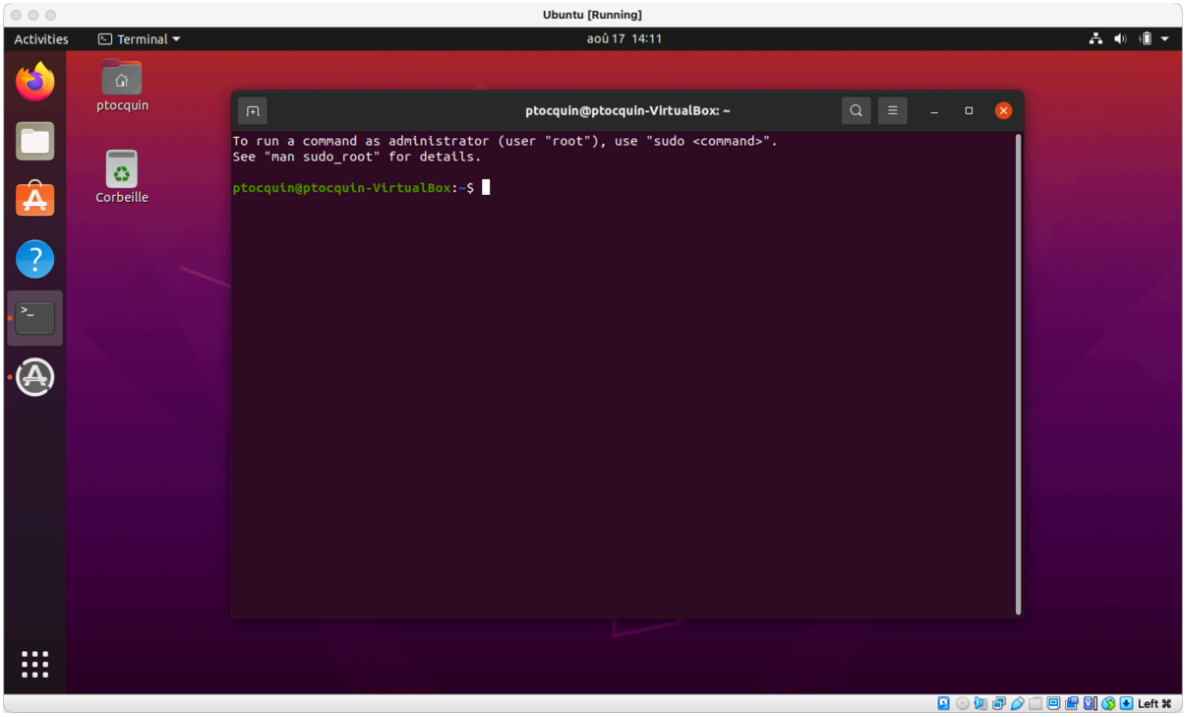

Figure 2.1: Le Terminal dans l'environnement Ubuntu 20.04. L'invite de commande est précédée par un message qui sera explicité plus loin.

traitement des noms de fichiers ou de répertoires qui contiennent des espaces (A éviter d'une manière générale !).

2.3.1 La commande

Dans la plupart des cas, le premier *mot* est interprété comme un nom de commande ou de programme. Nous verrons plus loin ce qui définit ce qu'est un programme pouvant être appelé à la ligne de commande. Lorsque la commande n'existe pas, un message d'erreur est retourné:

```
$ whatever you do, do it right
whatever: command not found
```

Vous le constatez, le message d'erreur ne fait référence qu'au premier *mot* et indique que cette commande n'est pas *trouvée*. Les autres *mots* ne sont donc pas interprétés.

Lorsque le premier mot correspond à une commande interprétable, elle est exécutée par le shell et elle produit son résultat qui peut se traduire par un affichage dans le terminal.

```
$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
```

Si vous comparez ce résulat à l'affichage de la Figure 1.2, vous comprendrez certainement que la fonction de la commande `ls` est d'afficher le contenu (fichiers et répertoires) d'un répertoire.

2.3.2 Les arguments et options

Cette première commande est un exemple qui illustre la philosophie Unix que nous avons rappelée plus haut: chaque commande réalise une tâche précise, mais elle le fait bien. Dans le cas de `ls`, *le faire bien* peut sembler un objectif peu ambitieux, mais ce qui se cache vraiment derrière ce mantra est que chaque commande doit pouvoir produire un résultat utile, une information complète ou une action subtile. En l'occurence ici, lister le contenu d'un répertoire peut impliquer de vouloir les trier par nom ou par taille, d'en afficher des informations sur leur nature, leur date de création ou de modification, etc… Il est donc généralement possible de moduler la manière dont une commande fonctionne (ce qu'elle produit comme résultat) en ajoutant des *mots* supplémentaires à la suite de la commande elle-même. Ils constituent alors des **arguments** de celle-ci et sont donc interprétés par le shell.

A partir de maintenant, vous êtes encouragés à tester vous-mêmes les commandes proposées comme exemples. Leur résultat ne sera plus systématiquement illustré.

```
$ ls -r
$ ls -a
$ ls /
$ ls -r ~
```

Les arguments `-r` et `-a` sont des **options** de la commande `ls`. Ils se distinguent des autres arguments (ici `/` et `~`) par le fait qu'ils commencent par un tiret simple `-` (ou un double tiret `--`).

2.3.3 Décrypter les commandes

Les options et les autres arguments qui sont acceptés par chaque commande sont strictement définis et sont consultables dans l'aide que le shell fournit via les commandes `man`, `info` ou l'option `--help` disponible pour la plupart des commandes.

```
$ man ls
$ info ls
$ ls --help
```

`man` et `info` sont des commandes comme toutes les autres. Elles acceptent des options et des arguments. Elles sont utilisées ici avec la chaîne de caractères `ls` comme argument pour spécifier quelle page des manuels doit être affichée. Pour vous en convaincre, n'hésitez pas à consulter l'aide relative à ces commandes.

```
$ man man
$ man info
```

Attardons nous sur le manuel de la commande `ls` (Figure 2.2).

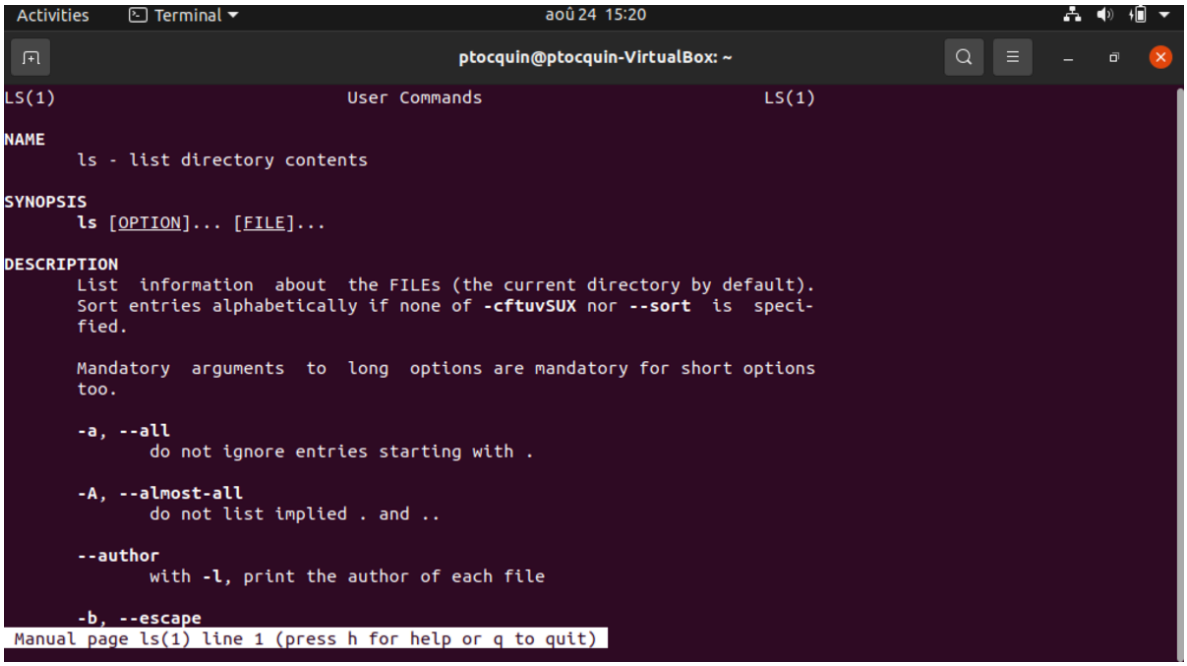

Figure 2.2: Manuel de la commande `ls`

Bibliographie

Barnes, H. (2021). Pro Windows Subsystem for Linux (WSL): Powerful Tools and Practices for Cross-Platform Development and Collaboration (Berkeley, CA: Apress).

McIlroy, M.D., Pinson, E.N., and Tague, B.A. (1978). UNIX Time-Sharing System: Forward. Bell System Technical Journal *57*, 1899–1904.