Single-Cell Transcriptomics : Analyse, Visualisation et Clustering avec R et Seurat

Auteur : Bioinformaticien en formation
Master de Bioinformatique, Université de Montpellier

November 27, 2025

Abstract

Cette revue méthodologique synthétise l'ensemble des concepts et applications pratiques liés à l'analyse de transcriptomes cellulaires uniques (*single-cell RNA-seq*). Elle couvre depuis la dissociation cellulaire et les biais techniques jusqu'aux méthodes avancées de réduction de dimension et de clustering, en intégrant des exemples concrets en `R` avec le package `Seurat`. Cette présentation vise à fournir une compréhension scientifique de haut niveau, pédagogique et didactique, adaptée à une portée internationale.

1 Introduction à la single-cell transcriptomics

La transcriptomique *single-cell* (*scRNA-seq*) permet d'explorer la diversité transcriptionnelle des cellules d'un tissu à l'échelle individuelle. Contrairement au RNA-seq classique, cette approche capture l'hétérogénéité cellulaire, identifie des sous-types rares, et révèle des trajectoires de différenciation. Les données scRNA-seq sont caractérisées par :

- Une très faible quantité d'ARN par cellule, nécessitant amplification enzymatique.
- Des biais techniques introduits lors de l'amplification ou du séquençage.
- Des effets de dropouts, où certains gènes exprimés ne sont pas détectés.

1.1 Exemple : PBMC

Les cellules mononucléées périphériques (*PBMC*) sont un modèle classique. Chaque cellule est séquencée pour obtenir son transcriptome individuel. Le clustering de ces cellules révèle des populations distinctes (T, B, NK, monocytes), tandis que les projections 2D (t-SNE, UMAP) permettent une visualisation spatiale intuitive de la similarité transcriptionnelle.
Remarque : la PCA est souvent utilisée comme prétraitement, non comme projection finale, pour réduire la dimensionnalité initiale avant t-SNE ou UMAP.

2 Prétraitement et contrôle qualité

2.1 Création de l'objet Seurat et calcul des métriques QC

Après chargement des données, il est essentiel de calculer des métriques de qualité pour filtrer les cellules aberrantes.

```r
library(Seurat)
library(dplyr)
library(patchwork)

# Chargement des donn es 10x Genomics
pbmc.data <- Read10X(data.dir="PBMC-10x/")
pbmc <- CreateSeuratObject(counts = pbmc.data, project="PBMC3K",
                           min.cells = 3, min.features = 200)

# Calcul du pourcentage de g nes mitochondriaux
pbmc[["percent.mt"]] <- PercentageFeatureSet(pbmc, pattern="^MT-")
```

```
12
13 # Visualisation QC : violin plots
14 VlnPlot(pbmc, features=c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol=3)
```

Listing 1: Création d'un objet Seurat et calcul des métriques QC

Les cellules présentant un nombre excessif de gènes détectés ou un pourcentage élevé de mitochondriaux peuvent être exclues afin de limiter les biais expérimentaux.

2.2 Filtrage des cellules

```
1 pbmc[["reliable"]] <- pbmc[["nCount_RNA"]] <= 7500 &
2                       pbmc[["percent.mt"]] <= 7
3 pbmc <- subset(pbmc, subset=reliable)
```

Listing 2: Filtrage des cellules selon QC

3 Normalisation et sélection des gènes variables

La normalisation vise à corriger les différences de profondeur de séquençage entre cellules. La détection des gènes les plus variables est cruciale pour identifier les marqueurs discriminants.

```
1 pbmc <- NormalizeData(pbmc)
2 pbmc <- FindVariableFeatures(pbmc, selection.method = "vst", nfeatures=2000)
3 top15 <- head(VariableFeatures(pbmc), 15)
4 VariableFeaturePlot(pbmc) %>% LabelPoints(points=top15, repel=TRUE)
```

Listing 3: Normalisation et sélection des gènes variables

4 Réduction de dimension et projections

4.1 Analyse en composantes principales (PCA)

La PCA transforme les données initiales en composantes orthogonales maximisant la variance.

```
1 all.genes <- rownames(pbmc)
2 pbmc <- ScaleData(pbmc, features = all.genes)
3 pbmc <- RunPCA(pbmc, features = VariableFeatures(pbmc), npcs = 30)
4 ElbowPlot(pbmc)
```

Listing 4: PCA sur les gènes variables

4.2 t-SNE

Le t-SNE conserve les relations locales, révélant la structure fine des sous-populations cellulaires. La perplexité doit être ajustée selon le nombre de cellules.

```
1 pbmc <- RunTSNE(pbmc, dims=1:10, perplexity=30)
2 DimPlot(pbmc, reduction="tsne", label=TRUE)
```

Listing 5: Projection t-SNE

4.3 UMAP

UMAP capture à la fois la structure locale et globale. Il est particulièrement adapté à des données complexes avec plusieurs sous-populations.

```
1 pbmc <- RunUMAP(pbmc, dims=1:10)
2 DimPlot(pbmc, reduction="umap", label=TRUE)
```

Listing 6: Projection UMAP

5 Clustering des cellules

5.1 Clustering basé sur les graphes (Louvain)

La construction d'un graphe de voisinage permet de détecter des communautés correspondant à des populations cellulaires.

```
pbmc <- FindNeighbors(pbmc, dims=1:10)
pbmc <- FindClusters(pbmc, resolution=0.5)
```

Listing 7: Clustering Louvain

5.2 Clustering hiérarchique et HDBSCAN

Pour des données avec des densités variables, HDBSCAN fournit une alternative robuste.

```
library(dbscan)
hdb <- hdbscan(pbmc@reductions$pca@cell.embeddings[,1:10], minPts=5)
```

Listing 8: HDBSCAN en R

6 Identification des marqueurs et annotation des populations

6.1 Détection de gènes spécifiques

```
pbmc.markers <- FindAllMarkers(pbmc, only.pos=TRUE, min.pct=0.25,
                               logfc.threshold=0.25)
pbmc.markers %>%
  group_by(cluster) %>%
  slice_max(n=2, order_by=avg_log2FC)
```

Listing 9: Identification des marqueurs par cluster

6.2 Visualisation des marqueurs

```
top10 <- pbmc.markers %>%
  group_by(cluster) %>%
  top_n(n=10, wt=avg_log2FC)
DoHeatmap(pbmc, features=top10$gene) + NoLegend()

FeaturePlot(pbmc, features=c("MS4A1","CD3E","GNLY","CD14","LYZ","PPBP"))
```

Listing 10: Heatmaps et FeaturePlots

6.3 Annotation experte

Sur la base des marqueurs connus et de la littérature, chaque cluster peut être annoté :

```
new.cluster.ids <- c("Naive␣CD4␣T", "CD14+␣Mono", "Memory␣CD4␣T", "B",
                     "CD8␣T", "FCGR3A+␣Mono", "NK", "DC", "Platelet")
names(new.cluster.ids) <- levels(pbmc)
pbmc <- RenameIdents(pbmc, new.cluster.ids)
DimPlot(pbmc, reduction="umap", label=TRUE, pt.size=0.5) + NoLegend()
```

Listing 11: Renommer les clusters avec les types cellulaires

7 Visualisations avancées

7.1 Distribution de l'expression génique

Ridgeplots et violin plots permettent de comparer l'expression des gènes entre clusters.

```
features <- c("LYZ","CD74","CST3","CD3D","FCGR3A","GZMA")
RidgePlot(pbmc, features=features, ncol=2)
VlnPlot(pbmc, features=features[c(1,2,4,6)])
```

Listing 12: Ridgeplots et violin plots

7.2 DotPlot et FeaturePlot avancé

```
DotPlot(pbmc, features=features) + RotatedAxis()
FeaturePlot(pbmc, features=c("MS4A1","PTPRCAP"), min.cutoff="q10", max.cutoff="q90")
```

Listing 13: DotPlot et FeaturePlot avancé

8 Sauvegarde et réutilisation des objets Seurat

```
saveRDS(pbmc, file="pbmc_final.rds")
```

Listing 14: Sauvegarde de l'objet Seurat

9 Perspectives et extensions

- **Intégration multi-omique** : combiner RNA-seq, protéomique et épigénomique pour une vue complète.
- **RNA velocity** : prédire les trajectoires dynamiques cellulaires à partir des transcrits pré-maturés.
- **Méthodes probabilistes** : scVI, scANVI pour gérer le batch effect et améliorer la détection de sous-populations rares.
- **Applications cliniques** : identification de sous-types tumoraux, réponse immunitaire, et médecine personnalisée.

10 Conclusion

Cette synthèse fournit un cadre complet pour l'analyse single-cell en R, depuis le prétraitement jusqu'à l'interprétation biologique. Chaque étape est commentée et justifiée, avec des exemples concrets pour PBMC. L'approche intégrée permet d'obtenir des résultats robustes, reproductibles et interprétables, tout en mettant en lumière l'hétérogénéité cellulaire et les sous-populations rares.