

Biyoinformatik · Uygulama notları

Tek Hücre Transkriptomiği

Damlacıklardan ve Barkodlardan Hücre Tiplerine,
Durumlarına, Diferansiyel İfadeye ve Şekillere

Damlacık temelli tek hücre RNA dizileme (scRNA-seq) için, kütüphane yapısı ve sayım matrisi üretiminden kalite kontrolüne, normalleştirmeye, keşif amaçlı gösterimlere, gözden geçirilmiş hücre tipi etiketlerine ve örnek yapısını gözeten çıkarıma uzanan pratik rehber. Temel örnek bağımsız donörler kullanır ve örnek düzeyinde toplulaştırılmış ham sayımlarla pseudobulk diferansiyel ifade analizine olanak tanır. Cell Ranger, STARsolo ve alevin-fry alternatif sayım yollarıdır; logaritmik normalleştirme, SCTransform ve scran farklı normalleştirme yaklaşımlarını gösterir. Bütünleştirme, düzeltilmemiş yapı ve ilgili biyolojiyle karşılaştırılarak değerlendirilen isteğe bağlı bir karardır. Ek bölümler göreli bileşimi, yörünge çıkarımının ve RNA velocity analizinin varsayımlarını, şekilleri, çekirdekleri ve büyük veri kümelerini ele alır. Okunabilir kodlar, çıkarım ve şekillere uzanan R yolu ile küme ve belirteçlere uzanan Python keşif yolu sağlar. İsteğe bağlı örnekler ve doğrulama sınırları temel çalıştırma sözleşmesinden ayrılır.

Kod ve yazılım ortamları:
github.com/tunabirgun/bioinformatics-notes

Tuna Birgun
github.com/tunabirgun

Orijinal Not: 15 Ocak 2026 · Türkçe: 29 Ekim 2026

İçindekiler

1 Genel bakış ve kapsam **3**

2 Bu çalışma notu nasıl kullanılır? **3**

3 İş akışına genel bakış **4**

4 Yazılımlar ve paketler **5**

5 Çalışma ortamının kurulumu **7**
- 5.1 İşletim sistemi ve kaynaklar . . . 7
- 5.2 Proje dizinleri ve örnek üst verisi . . . 7
- 5.3 Sayım ve Python ortamları . . . 7
- 5.4 R ve Bioconductor . . . 7

6 Okumalardan sayım matrisi üretimine **8**
- 6.1 Okumaları örneklere ayırma İsteğe bağlı . . . 8
- 6.2 Cell Ranger count . . . 8
- 6.3 Alternatif: STARsolo . . . 9
- 6.4 Alternatif: alevin-fry . . . 9
- 6.5 Standartlaştırılmış matrisi okuyun . . . 10

7 Alternatif başlangıç noktaları **10**
- 7.1 10x sayım matrisinden başlama . . . 11
- 7.2 Mevcut analiz nesnesinden başlama . . . 11
- 7.3 Herkese açık veriler . . . 12

8 Kalite kontrolü ve hücre çağırma **12**
- 8.1 Ham damlacıklardan hücre çağırma İsteğe bağlı . . . 12
- 8.2 Ortam RNA'sı değerlendirmesi İsteğe bağlı . . . 13
- 8.3 Hücre başına ölçütleri inceleyip filtre seçme . . . 14
- 8.4 Doublet saptama . . . 14

9 Normalleştirme ve özellik seçimi **15**
- 9.1 Bir normalleştirme yolu seçin . . . 15
- 9.2 Yüksek değişkenlik gösteren genlerin seçimi . . . 17
- 9.3 Ölçekleme ve istenmeyen değişkenlik . . . 17

10 Boyut indirgeme ve bütünleştirme **18**
- 10.1 PCA . . . 18
- 10.2 Bütünleştirme gerekip gerekmediğine karar verme . . . 18
- 10.3 Bütünleştirme alternatifleri . . . 19
- 10.4 Çizge kurma ve görselleştirme . . . 20

11 Kümeleme ve hücre tipi anotasyonu **20**
- 11.1 Çizge temelli kümeleme . . . 20
- 11.2 Betimleyici belirteç genler . . . 21
- 11.3 Referans temelli etiketleme . . . 22
- 11.4 İncelenmiş etiketleri doğrulayıp saklama . . . 22

12 Koşullar arasında diferansiyel gen ifadesi **23**
- 12.1 Pseudobulk: Ham sayımları örnek içinde toplulaştırın . . . 23
- 12.2 Havuzlanmış hücre başına testler koşul çıkarımında neden başarısız olur? . . . 25
- 12.3 Diferansiyel bolluk: Göreli bileşim . . . 26

13 Yörünge ve RNA velocity İsteğe bağlı **26**
- 13.1 Yörünge çıkarımı ve göreli süreç sıralaması . . . 27
- 13.2 RNA velocity . . . 27

14 Tek hücre verileri için şekiller **28**

14.1 Hücre tipi ve koşula göre UMAP 29
14.2 Belirteç ifade haritaları 29
14.3 Kanonik belirteçler için nokta grafikleri ve keman grafikleri 29
14.4 Örnekler arasında hücre tipi oranları 30
14.5 Pseudobulk analizi: volkan grafiği ve ısı haritası 30

15 Tamamlayıcı bilgiler **31**
15.1 Çok örnekli iş akışı 31
15.2 Tek çekirdek RNA dizileme 32
15.3 Büyük veri kümelerine ölçekleme 32
15.4 Yeniden üretilebilirlik 33
15.5 Sorun giderme 34

Kaynaklar **35**

1 Genel bakış ve kapsam

Tek hücre RNA dizileme (scRNA-seq), yakalanmış hücreleri ayrı ayrı profiller; ortalama ifade farklarının yanında hücre tipi bileşiminin ve doku içi değişkenliğin incelenmesini sağlar [1, 2]. Ortaya çıkan veri, RNA bolluğunun gürültülü ve eksik bir ölçümüdür; bir barkod tek başına canlı bir tek hücreyi veya ayrı bir biyolojik kimliği kanıtlamaz.

Bu not, damlacık temelli 10x Genomics Chromium 3′ gen ifadesi sayımı üzerinde durur. Barkodlar damlacıkları tanımlar; benzersiz moleküler tanımlayıcılar (UMI), aynı yakalanmış molekülün çoğaltılmış gözlemlerini birleştirmeye yardımcı olur. UMI temelli yinelenen gözlemleri birleştirme, atanan gene ve sayım algoritmasına da bağlıdır [4, 5, 6].

Ele alınan sorular. İş akışı, sayım matrisinden hücre tiplerinin tanımlanmasına ve koşulların karşılaştırılmasına ilerler. Hücre içeren damlacıklar belirlenip filtrelenir; iki hücrenin birlikte yakalanmasıyla oluşan doublet gözlemleri araştırılır. Keşif amaçlı gösterim, hücre gruplarını incelemeye yardımcı olur. Hücre tipi etiketleri gözden geçirildikten sonra koşullar arasındaki ifade veya göreli bileşim karşılaştırılır. Bütünleştirme, yörünge ve RNA velocity örnekleri ise ek varsayımları olan isteğe bağlı analizlerdir.

Kapsam dışındaki sorular. Standart 3′ UMI sayımı, tam uzunluklu transkript izoformlarının analizi, mekânsal ölçüm veya mutlak RNA ya da doku hücre sayısı ölçümü değildir. Toplu RNA dizileme (bulk RNA-seq) ile scRNA-seq arasındaki seçim; soruya, bağımsız tekrarlara, doku örneklemesine ve bütçeye bağlıdır. Hücre tipi bileşimi veya tip içi heterojenlik önemli olduğunda tek hücre verileri özellikle yararlıdır; bulk RNA-seq, tüm dokunun ifade karşılaştırması için yararlılığını korur.

Uygulama notu

Çalışma tasarımı. Temel örnek, insan 10x 3′ v3 verilerini, iki koşulu ve bağımsız donörleri kullanır; her donör için bir damlacık yakalama işlemi (capture) ve bir kütüphane vardır. Örnek tablosu `sample` , `condition`, `donor` ve `capture` alanlarını taşır. Koşul başına üç örnek, donörlere ait dosya düzenini gösterir; gerekli tekrar sayısı bilimsel soruya ve beklenen değişkenliğe göre belirlenmelidir. Eşleştirilmiş, tekrarlı, çoklanmış (birden fazla örneğin birlikte işlendiği) veya teknik tekrar içeren tasarımlar, çıkarım öncesinde açık bir genişletme gerektirir.

Bağımsız birimler. Aynı donörden gelen hücreler iç içe gözlemlerdir; ek bağımsız donör değildir. Koşul etkileri için örnek yapısını gözeten çıkarım kullanın ve donör/kütüphane/yakalama işlemi ilişkilerini süreç boyunca koruyun. Doublet saptama, işlem birimi olarak yakalama işlemini kullanır; bu birim ancak çalışma tasarımındaki varsayım altında örnekle aynıdır (Bölüm 8) [37].

Araçlar. Ana örnekler Seurat (R) [20] ile Scanpy/AnnData (Python) [21, 22] kullanır. Bioconductor'ın SingleCellExperiment çerçevesi ve OSCA iş akışı bazı aşamalar için alternatif uygulamalar sunar [3]. Örnekler benzer görevleri açıklar; veri gösterimleri denetlenmeden yolların sayısal olarak özdeş veya birbirinin yerine geçebilir olduğu varsayılmamalıdır.

2 Bu çalışma notu nasıl kullanılır?

Numaralı bir aşamayı çalıştırmadan önce deney tasarımını ve girdi sözleşmesini okuyun. Sayım gerekiyorsa FASTQ'dan başlayın; doğrulanmış filtrelenmiş bir matris kalite kontrolü (QC) aşamasına, mevcut bir nesne ise sayımları, işlemleri ve üst verisi anlaşıldıktan sonra daha ileri bir aşamaya girebilir (Bölüm 7). Bir sayım ve bir normalleştirme yolu seçin. Bütünleştirme, zorunlu bir düzeltme değil ayrı bir karardır.

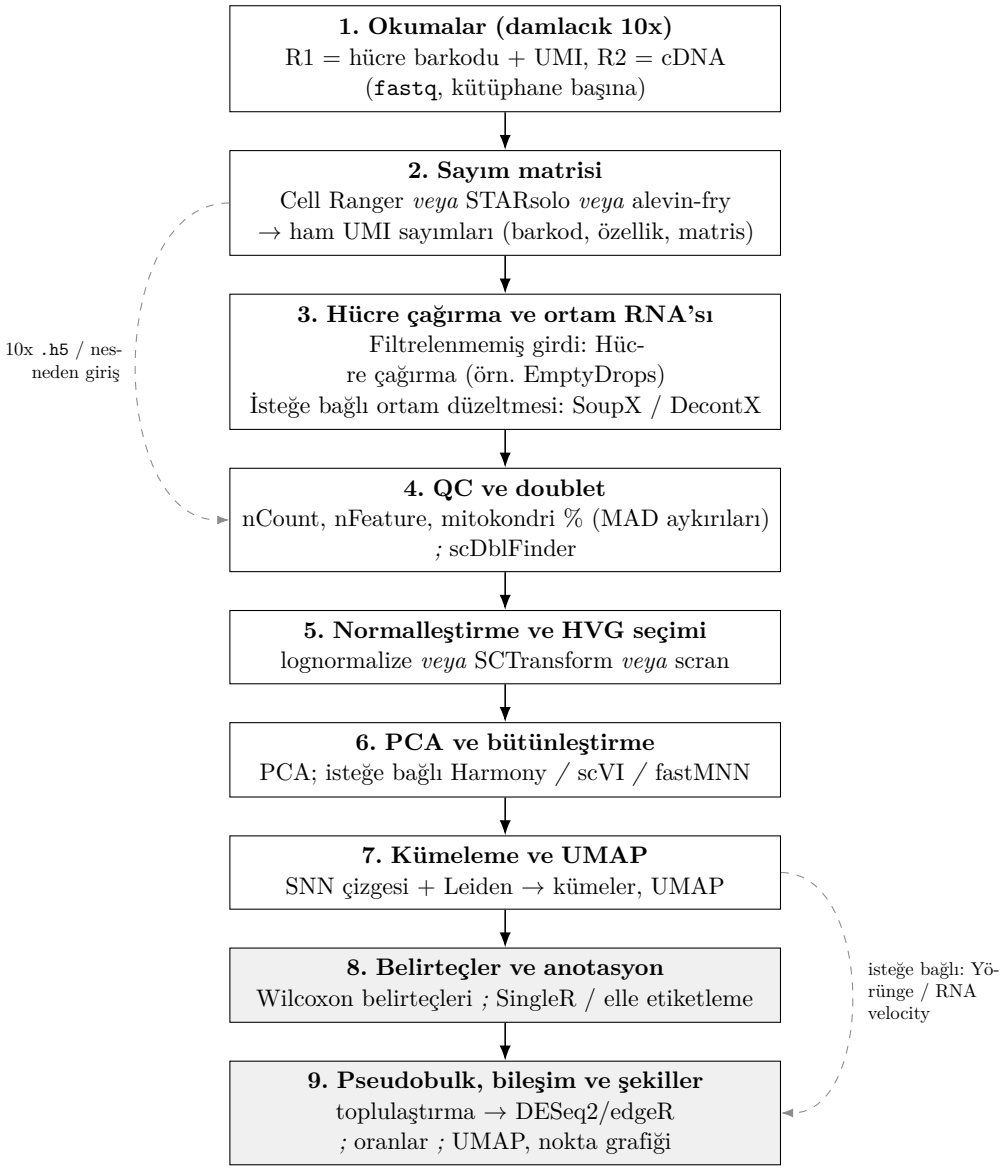
Kod kutuları SHELL, R veya PYTHON olarak etiketlenir. **Temel parametreler**, bilinçli seçilecek ayarları; **Girdiler ve çıktılar**, veri sözleşmesini; **Dikkat**, hata kaynaklarını; **İpucu**, pratik önerileri gösterir. İsteğe bağlı isteğe bağlı bir dalı belirtir. Çalıştırmadan önce `<sample>` gibi yer tutucuları değiştirin; açıklama amaçlı kodlar, ön koşullarında adı verilen nesneleri gerektirebilir.

Uygulama notu

Matris yönelimi. 10x Matrix Market dosyası ve Seurat RNA sayımları, satırlarda genleri, sütunlarda barkodları/hücreleri tutar. AnnData ise satırlarda hücreleri, sütunlarda genleri tutar. Aynı kavramsal ölçümler zıt yönelimlerle kodlanır. Ham UMI sayımları sonlu, negatif olmayan tam sayılardır; normalleştirilmiş ifadeden ayrı korunmalıdır.

Eşlik eden klasör okunabilir, numaralı kodlar sağlar. R yolu kalite kontrolünden hücre tipi anotasyonuna, hücrelere ait ham gen sayımlarını örnek düzeyinde toplulaştıran pseudobulk analizine, bileşim analizine ve temel şekillere kadar ilerler; Python yolu ise kalite kontrolünden keşif amaçlı kümeleme ve belirteç genlerin belirlenmesine kadar gider. İsteğe bağlı EmptyDrops, ortam RNA'sı düzeltmesi, scVI, yörünge ve velocity örneklerinin kendi ön koşulları vardır; bu kodlar tarafından otomatik olarak çalıştırılmazlar. Komutlar ve doğrulama durumu için eşlik eden README dosyasına bakın: Sözdizimi ve sentetik sözleşme kontrolleri, biyolojik bir veri kümesinin yeniden üretilmesinden farklıdır.

3 İş akışına genel bakış



4 Yazılımlar ve paketler

Komut satırı araçları bioconda ve conda-forge kanallarıyla conda üzerinden kurulabilir; Cell Ranger, 10x Genomics'ten ayrı kurulur. R paketleri CRAN ve Bioconductor'dan, Python paketleri PyPI veya conda-forge üzerinden sağlanır. Bioconductor tek hücre iş akışı (OSCA), tamamlayıcı bir R/Bioconductor yaklaşımı sunar [3].

Komut satırı araçları. Aşağıdaki araçlar FASTQ verilerini genler×barkodlar UMI sayım matrisine dönüştürür veya bu işlemi destekler.

Araç	**Sürüm**	**Amaç**
Cell Ranger	—	10x FASTQ'dan sayım matrisi; 10x'ten ayrı kurulur [4, 7]
STARsolo	STAR 2.7.x ile	Alternatif sayım matrisi üretimi [8]
salmon + alevin-fry	—	Alternatif sayım; intronları çıkarılmış/korunmuş RNA için USA modu [9]
samtools	—	BAM/SAM dizinleme, sıralama ve inceleme
bcl2fastq / bcl-convert	—	BCL → FASTQ dönüşümü ve örneklere ayırma; Illumina

Uygulama notu

Kodlar sayım araçlarıyla Python analizini ayrı ortamlarda tutar (Bölüm 5); Cell Ranger, `PATH` değişkenine eklenen bağımsız bir dağıtımdır.

R / Bioconductor paketleri. Tablo, nesne yönetiminden bütünleştirme, hücre tipi anotasyonu ve pseudobulk diferansiyel ifade analizi aşamalarına uzanan R tarafını kapsar.

Paket	Amaç
Seurat	Temel nesne, kalite kontrolü, kümeleme, bütünleştirme ve diferansiyel ifade [20]
SingleCellExperiment / OSCA	Bioconductor nesne sınıfı ve uçtan uca iş akışı [3]
scran	Havuzlama/ayrıştırma ile normalleştirme ve varyans modelleme [13, 14]
scater	QC ölçütleri, aykırı değer saptama ve görselleştirme [15]
DropletUtils	10x matrislerini okuma ve `emptyDrops` ile hücre çağırma [10]
sctransform	Düzenlileştirilmiş negatif binom modeliyle normalleştirme ve varyans kararlılaştırma [23]
SoupX İsteğe bağlı	Ortam RNA'sı bulaşmasını tahmin etme ve düzeltme [11]
celda / DecontX İsteğe bağlı	Alternatif ortam RNA'sı düzeltmesi [12]
scDblFinder	Yapay doublet gözlemleri ile saptama [16]
DoubletFinder	Alternatif doublet saptama [17]
batchelor (fastMNN)	Karşılıklı en yakın komşularla bütünleştirme [25]
harmony	Teknik parti/düşük boyutlu gösterim bütünleştirmesi [24]
SingleR	Referans temelli otomatik hücre tipi etiketleme [34]
Azimuth	Seurat ekosisteminde referansa eşleme [35]
presto	Hızlı Wilcoxon sıra toplamı belirteç testleri [33]
muscat	Pseudobulk toplulaştırma ve koşullar arasında diferansiyel ifade [38]
DESeq2	Bulk/pseudobulk diferansiyel ifade [39]
edgeR	Bulk/pseudobulk diferansiyel ifade [40]
miloR	Komşuluk düzeyinde diferansiyel bolluk [41]
speckle / propeller	Hücre tipi oranları için diferansiyel bolluk [42]
slingshot İsteğe bağlı	Düşük boyutlu uzayda yörünge çıkarımı [43]
clusterProfiler İsteğe bağlı	Diferansiyel ifade gen listelerinin işlevsel zenginleşme analizi [48]

Python paketleri. Tablo, Scanpy/AnnData tabanlı araçları gösterir.

Paket	Amaç
scanpy	QC, normalleştirme, kümeleme ve ifade analizi arayüzü [21]
anndata	Scanpy'nin kullandığı açıklamalı veri matrisi [22]
scvi-tools	Derin üretici modelle bütünleştirme ve modelleme [27, 26]
harmonypy	Harmony bütünleştirmesinin Python uygulaması [24]
leidenalg / igraph	Komşuluk çizgesinde Leiden topluluk saptama [31]
scrublet	Yapay doublet simülasyonuyla saptama [49]
celltypist	Referans temelli otomatik hücre tipi etiketleme [36]
scVelo İsteğe bağlı	RNA velocity için dinamik model [46]
umap-learn	`sc.tl.umap` tarafından kullanılan UMAP yerleştirmesi [29]

Dikkat edilmesi gerekenler

Her analiz için çözülmüş ortamı, platformu, kanalları ve paket sürümlerini kaydedip kilitleyin. Sağlanan `environment.yml`, çözülmemiş bir manifesttir; kilit dosyası değildir. Paket sürümleri arasında sonuçlar değişebilir; yalnızca rastgelelik tohumu belirlemek bu farkı gidermez.

5 Çalışma ortamının kurulumu

Yazılım kurmadan önce gerekli yolu seçin. Sayım ve sayım sonrası analiz, farklı platform ve kaynak gereksinimlerine sahiptir; isteğe bağlı modelleme paketlerinin temel ortamda bulunması gerekmez. Sağlanan manifest dosyaları kurulum tarifleridir; çözülmüş, taşınabilir bağımlılık kilit dosyaları değildir.

5.1 İşletim sistemi ve kaynaklar

Cell Ranger, desteklenen Linux sistemleri için dağıtılır; sayım öncesinde sürüme özgü donanım ve işletim sistemi gereksinimlerini denetleyin [7]. Linux sunucusu veya HPC kaynağı yaygın bir seçenektir. WSL2, yeterli atanmış bellek ve uyumluluk kontrolleri gerektirir; kullanılabilir olması üretici desteğini göstermez. STARsolo ve salmon/alevin-fry araçlarının kendi platform sürümleri ve referans indeksi gereksinimleri vardır. Tek bir sayım ortamının tüm macOS mimarilerinde çözüleceğini varsaymayın.

Uyumlu paket sürümleri varsa R ve Python ile sayım sonrası analiz Linux, macOS veya Windows üzerinde çalışabilir. Bellek gereksinimi; hücre sayısına, özellik sayısına, ara gösterimlere ve algoritma seçimlerine bağlıdır. Büyük nesneleri yüklemeden veya paralel işçiler başlatmadan önce kullanılabilir RAM ve disk alanını inceleyin. Açıkça seçilmiş iş parçacığı sayısını hem işlemciye hem belleğe göre belirleyin; daha fazla iş parçacığı bir referans dizininin boyutunu küçültmez.

5.2 Proje dizinleri ve örnek üst verisi

Eşlik eden kaynak kodu, referansları, verileri ve sonuçları içeren analiz kökünden ayrı tutun. Aşağıdaki komutlar başlatılmış bir bash kabuğu varsayar; kod klasörünün yolunu gerçek konumuyla değiştirin.

Komut satırı

```
export SC_WORKFLOW="/path/to/main_scripts/singlecell_transcriptomics"
export SCRNA_PROJECT="$HOME/scrna_project"
export SCRNA_THREADS=8   # example allocation: choose for your machine
bash "$SC_WORKFLOW/scripts/00_setup/01_project_dirs.sh"
cd "$SCRNA_PROJECT"
```

Kurulum `ref/`, `fastq/`, `counts/`, `config/` ve `results/` dizinlerini oluşturur. `config/samples.tsv` şablonunu yalnızca dosya yoksa kopyalar. Analizden önce örnek tanımlayıcılarını ve koşulları gerçek tasarımla değiştirin. Temel kodlar her örnek için bağımsız bir donör ve bir yakalama işlemi varsaydığından, tekrarlanan donörleri veya yakalama işlemi değerlerini reddeder; eşleştirilmiş veya çoklanmış tasarımları uygulamaz.

5.3 Sayım ve Python ortamları

Komut satırı

```
# Python exploratory route; use a separate environment for optional packages.
conda env create -f "$SC_WORKFLOW/environment.yml"
conda activate scrna

# Alternative counting tools, normally on Linux; activate separately.
conda env create -f "$SC_WORKFLOW/environment-counting.yml"
# conda activate scrna-counting
```

Cell Ranger, 10x Genomics'ten ayrı kurulur ve `PATH` değişkenine eklenir; gerçek sürümünü, eşleşen referansı ve kütüphane kimyasını kaydedin. İsteğe bağlı scVI, CellTypist ve scVelo örneklerinin ek bağımlılıkları `requirements-optional.txt` dosyasında listelenir; doğrulanmış bir analiz ortamını incelemeden değiştirmek yerine bunları yalıtılmış bir ortamda çözün ve sınayın.

5.4 R ve Bioconductor

Temel R yolu Seurat, scDblFinder, SingleR/celldex, DESeq2 ve speckle kullanır; Harmony yalnızca isteğe bağlı bütünleştirme dalında gerekir. Kurucu bir SVG aygıtı da ister. İsteğe bağlı bölüm örnekleri, kendi kod kutularında adı verilen paketleri gerektirir.

Komut satırı

```
Rscript "$SC_WORKFLOW/R/install_packages.R"
```

Çözülen R ve Bioconductor sürümlerini ve paket bağımlılıklarını kaydedin. Sürdürülen bir analizde sınanmış bir kilit dosyası varsa onu geri yükleyin; yoksa uyumluluğu denetledikten sonra proje ortamı oluşturun ve kaydı koruyun. Ortam dışa aktarımı, tek bir platformdaki tek bir çözümü tanımlar; başka bir platformun aynı ortamı geri yükleyebileceğinin kanıtı değildir.

Dikkat edilmesi gerekenler

Tek hücre matrisinin tamamını seyrek tutun: R'de seyrek Matrix sınıfları, Python'da CSR/CSC dizileri kullanın. Ölçekleme, model uyarlama veya biçimler arası dönüşüm yine de yoğun ara yapılar oluşturabilir. Küçük bir pseudobulk matrisini yoğun biçime dönüştürmek, binlerce hücreyi içeren matrisi yoğunlaştırmaktan farklıdır. Rastlantısal işlemlerin yakınında rastgelelik tohumu belirleyin ve sürümleri kaydedin; tohumlar yazılım sürümleri, donanım veya algoritma uygulamaları arasında özdeş sonuçları garanti etmez.

6 Okumalardan sayım matrisi üretimine

Sayım, cDNA okumalarını genlere atar, hücre barkodlarını ve UMI'leri düzeltir ve barkod/gen grupları içindeki yinelenen gözlemleri birleştirir. Çalışma örneğindeki 10x 3′ v3 yapısında R1, 16 bazlık hücre barkodu ve 12 bazlık UMI taşırken R2 cDNA taşır. Diğer kütüphane kimyası seçenekleri farklı düzenler kullanır; dosya adından çıkarmak yerine kütüphanenin belgelenmiş geometrisini esas alın [4, 7].

6.1 Okumaları örneklere ayırma İsteğe bağlı

Girdi bir dizileyicinin BCL çıktısıysa desteklenen Illumina BCL Convert iş akışı ve doğru indeks/örnek tablosuyla örnek FASTQ dosyalarını oluşturun. Mevcut `cellranger mkfastq` iş akışları bcl2fastq aracını sarmalar; ancak 10x artık bu komut yerine BCL Convert önerir ve komutu kullanımdan kaldırma sürecindedir [7]. Okumaları örneklere ayırırken barkod veya UMI dizisini kırpmayın. Bir dizileme merkezi FASTQ sağlıyorsa sayımdan önce okuma eşlerini, örnek atamasını ve kütüphane kimyasını doğrulayın.

6.2 Cell Ranger count

Cell Ranger okumaları hizalar, barkod/UMI işlemlerini yürütür, hücre çağırma yapar ve ham/filtrelenmiş matrisler ile kütüphane başına QC raporu yazar. Bu örnekte her bağımsız kütüphane için bir kez çalıştırın; aynı kütüphanenin dizileme hatları, aynı sayım çalışmasının girdileridir ve biyolojik tekrar değildir.

Komut satırı

```
cellranger count --id=sample1 \
  --transcriptome=refdata-gex-GRCh38-2024-A \
  --fastqs=fastq/ --sample=sample1 --create-bam=true
```

Girdiler ve çıktılar

Girdiler: Uyumlu FASTQ dosyaları ve tür/genom sürümüyle eşleşen 10x referansı. `sample1/outs/` altındaki çıktılar: Çağrılmış hücreler için `filtered_feature_bc_matrix/`, çağrılmamış damlacıkları da içeren `raw_feature_bc_matrix/` ve `web_summary.html`. Matrix Market sayımları genler×barkodlar yönelimindedir. İlerlemeden önce eşleme, dizileme doygunluğu, hücre çağırma ve bulaşma tanılarını inceleyin.

Temel parametreler

Gerçek Cell Ranger sürümünü, kimyayı, referans ve gen anotasyonu sürümü bilgisini, FASTQ seçimini ve ilgili sayım seçeneklerini kaydedin. Cell Ranger 7 ve sonraki sürümler intronik okumaları varsayılan olarak içerir; seçilen sürümün ayarını doğrulayın ve örnekler arasında tutarlı tutun. Hücre çağırma parametreleri ölçüme ve gözlenen verilere bağlıdır; yalnızca istenen hücre sayısına ulaşmak için değiştirmeyin.

Eşlik eden `01_cellranger_count.sh`, çalışma dizinlerini analiz kökündeki `count_runs/` altında tutar ve yalnızca filtrelenmiş üçlü dosya grubunu `counts/<sample>/` altına kopyalar. Mevcut bir hedefi reddeder. Hücre çağırma veya ortam RNA'sı çalışmaları için özgün ham çıktıyı koruyun; kopyalanmış filtrelenmiş dizin onun yerini tutmaz.

6.3 Alternatif: STARsolo

STARsolo, STAR içinde barkod/UMI sayımı yapar [8]. Cell Ranger ile karşılaştırmada referansın, okuma geometrisinin, UMI kurallarının, intron yaklaşımının ve hücre çağırma ayarlarının eşleşmesi önemlidir; eşleme aracı değişikliği sayısal özdeşliği garanti etmez.

Komut satırı

```
STAR --runMode alignReads --genomeDir star_index \
  --soloType CB_UMI_Simple --soloCBwhitelist 3M-february-2018.txt \
  --soloCBstart 1 --soloCBlen 16 --soloUMIstart 17 --soloUMIlen 12 \
  --soloFeatures GeneFull --soloCellFilter EmptyDrops_CR \
  --readFilesIn R2.fastq.gz R1.fastq.gz --readFilesCommand zcat \
  --outFileNamePrefix sample1_
```

STAR önce cDNA, sonra barkod/UMI dizisi beklediği için `--readFilesIn` içinde R2, R1'den önce gelir. Kod, hat dosyalarını açıkça eşleştirir. `GeneFull` intronik sinyali içerir; `Gene` ise farklı bir hedefe sahip ekzon temelli alternatiftir. STAR dizinini ve anotasyonu amaçlanan karşılaştırmaya göre doğrulayın. V2 barkod listesi/UMI düzeni, v3 yerine kullanılamaz.

Yukarıdaki bağımsız komut, çalışma dizininde `sample1_Solo.out/` üretir. Aynı analiz kökünden filtrelenmiş üçlü dosya grubunu eşlik eden uyarlayıcıyla bir kez `counts/sample1/` altına aktarın:

Komut satırı

```
Rscript "$SC_WORKFLOW/scripts/01_count_matrix/04_import_matrix.R" \
  10x sample1_Solo.out/GeneFull/filtered sample1
```

6.4 Alternatif: alevin-fry

salmon alevin, okumaları splici (intronları çıkarılmış transkriptlerle intronik dizileri birleştiren referans) üzerine eşleyebilir; bunu alevin-fry ile izinli barkod listesinin üretilmesi, kayıtların gruplanması ve nicelendirme aşamaları izler [9]. Dizin, transkript–gen eşlemesi ve gen sembolü eşlemesi aynı anotasyondan gelmelidir. Örneği çalıştırmadan önce bunları seçilen sürümün belgelenmiş referans iş akışıyla oluşturun; tek başına bir dizin yolu, eksiksiz referans sözleşmesi değildir.

Komut satırı

```
salmon alevin -i salmon_splici -l ISR \
  -1 R1.fastq.gz -2 R2.fastq.gz --chromiumV3 --sketch -o map
alevin-fry generate-permit-list -i map -d fw -o quant \
  --unfiltered-pl 3M-february-2018.txt
alevin-fry collate -i quant -r map -t 8
alevin-fry quant -i quant -m t2g_3col.tsv -r cr-like \
  -o quant/res --use-mtx -t 8
```

Üç sütunlu transkript/gen/durum eşlemesi, USA nicelendirmesini etkinleştirir. Buradaki durumlar intronları korunmuş (unspliced, U), intronları çıkarılmış (spliced, S) ve intron durumu belirsiz (ambiguous, A)

olarak tanımlanır. Velocity incelenecekse bu durumları koruyun. Bunlar olağan bir 10x filtrelenmiş matris değildir: USA yapısını anlayan okuyucu kullanın, ifade gösterimini açıkça seçin ve barkod düzeltmesini hücre çağırmadan ayırın.

Kodun içe aktarma dalı `fishpond::loadFry(outputFormat="all")` kullanarak U+S+A değerlerini gen başına bir kez toplar; özgün USA dosyalarını ayrı tutar. Sayım anotasyonundan gelen benzersiz `gene_id` ve `gene_symbol` sütunlu `config/gene_symbols.tsv` dosyası ile her satırda bir çağrılmış barkod içeren gözden geçirilmiş listeyi gerektirir. Bu listeyi sağlamadan önce uygun yöntemle ham barkod matrisi üzerinde hücre çağırma yapın (Bölüm 8).

Bağımsız komutlar `quant/res` üretir; eşlik eden sayım betiği ise `counts/<sample>_quant/res` yolunu kullanır. Seçtiğiniz yolun ürettiği konumu verin. Yukarıdaki bağımsız komutlar için:

Komut satırı

```
Rscript "$SC_WORKFLOW/scripts/01_count_matrix/04_import_matrix.R" \
  fry quant/res sample1 config/sample1_called_barcodes.tsv
```

6.5 Standartlaştırılmış matrisi okuyun

Kodun sözleşmesi, `counts/<sample>/` altında sıkıştırılmış `matrix.mtx.gz`, `features.tsv.gz` ve `barcodes.tsv.gz` dosyalarıdır. Okuyucu matris boyutlarını, sayım değerlerini ve tanımlayıcıları denetler, Gene Expression özelliklerini seçer ve görüntüleme sembolleriyle birlikte kararlı gen kimliklerini korur. Örnekler arasında referans/özellik sırası aynı olmalıdır.

R

```
source(file.path(Sys.getenv("SC_WORKFLOW"), "R/contracts.R"))
input <- read_counts("counts/sample1")
so <- Seurat::CreateSeuratObject(input$counts, min.cells = 0, min.features = 0)
# Alternative for an original raw 10x directory:
sce <- DropletUtils::read10xCounts("sample1/outs/raw_feature_bc_matrix")
```

PYTHON

```
import scanpy as sc
adata = sc.read_10x_mtx("counts/sample1", var_names="gene_symbols")
adata.var_names_make_unique()
```

Dikkat edilmesi gerekenler

Filtrelenmiş matrisler zaten hücre çağırmadan geçmiştir; bu kısıtlanmış kümede EmptyDrops'u yeniden çalıştırmayın. Ortam profiline dayalı hücre çağırma testi veya SoupX arka plan tahmini için özgün ham matrisi kullanın. İntron atamaları, gen anotasyonları veya barkod/UMI kuralları farklıysa boyutlar ve dosya adları aynı olsa bile sayımlar değişebilir. İçe aktarma uyarlayıcısı biçimin korunduğunu denetler; eşleme aracı eşdeğerliğini veya hücre kimliğini doğrulamaz.

7 Alternatif başlangıç noktaları

Bu bölüm üç başlangıç seçeneğini ele alır: Bir sayım aracının ürettiği sayım matrisi, kaydedilmiş analiz nesnesi veya herkese açık veri kümesi. Ham USA matrisi kendi okuma ve hücre çağırma yolunu gerektirir; otomatik olarak filtrelenmiş 10x matrisi değildir. Elinizdeki girdiye uyan seçeneği belirleyin ve kalite kontrolü (Bölüm 8) veya doğrulanmış daha ileri bir aşamadan başlayın.

7.1 10x sayım matrisinden başlama

10x `filtered_feature_bc_matrix` dizini ile tek dosyalı `.h5` karşılığı, genler×barkodlar yönelimindeki aynı seyrek UMI sayımlarını kodlar. Seurat her iki biçimi de okur. Bioconductor, dizin biçimindeki veriyi `SingleCellExperiment` nesnesine aktarır.

R

```r
library(Seurat)

# from the .h5 file (single file, convenient for sharing)
m <- Read10X_h5("sample1/outs/filtered_feature_bc_matrix.h5")
if (is.list(m)) m <- m[["Gene Expression"]]
so <- CreateSeuratObject(m, project = "sample1", min.cells = 0, min.features = 0)

# from the mtx/ directory (barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz)
m <- Read10X("sample1/outs/filtered_feature_bc_matrix")
if (is.list(m)) m <- m[["Gene Expression"]]
so <- CreateSeuratObject(m, project = "sample1", min.cells = 0, min.features = 0)

# OSCA / Bioconductor entry point
library(DropletUtils)
sce <- read10xCounts("sample1/outs/filtered_feature_bc_matrix", col.names = TRUE)
```

PYTHON

```python
import scanpy as sc

# from the .h5 file
adata = sc.read_10x_h5("sample1/outs/filtered_feature_bc_matrix.h5")
adata.var_names_make_unique()

# from the mtx/ directory
adata = sc.read_10x_mtx("sample1/outs/filtered_feature_bc_matrix", var_names="gene_symbols")
```

Girdiler ve çıktılar

Girdi: Filtrelenmiş veya ham bir 10x `outs/` dizini ya da `.h5` dosyası. Çıktı: Ham UMI sayımlarını `counts/so[["RNA"]]$counts/adata.X` içinde tutan bir `Seurat`, `SingleCellExperiment` veya `AnnData` nesnesi; henüz normalleştirme uygulanmamıştır. Gösterilen kurucular içe aktarılan özellikleri ve hücreleri korur; QC ve üst veri ataması ayrı adımlardır.

7.2 Mevcut analiz nesnesinden başlama

Bir çalışma arkadaşının kaydettiği `.rds` Seurat veya `.h5ad` AnnData nesnesi doğrudan yüklenebilir ya da başka bir ekosistemde çalışmak üzere dönüştürülebilir. `sceasy` ve `SeuratDisk`, Seurat↔AnnData dönüşümü; `zellkonverter`, SingleCellExperiment↔AnnData dönüşümü sağlar.

R

```r
library(Seurat)
so <- readRDS("sample1_processed.rds")

# convert to AnnData for a Python session (sceasy)
library(sceasy)
sceasy::convertFormat(so, from = "seurat", to = "anndata", outFile = "sample1.h5ad")
# Version-dependent demonstration: verify v5 layer support before using it.

# SingleCellExperiment <-> AnnData (zellkonverter)
library(zellkonverter)
sce <- readH5AD("sample1.h5ad")
```

PYTHON

```python
import anndata as ad

adata = ad.read_h5ad("sample1_processed.h5ad")
```

Dikkat edilmesi gerekenler

Dönüşüm sonrasında hem değerler hem kurallar doğrulanmalıdır: Seurat'ın `scale.data` alanı, v5 veri katmanları yapısı ve AnnData'nın `.obsm`/`.uns` alanları bire bir eşleşmez. İlerlemeden önce ham sayımların, normalleştirilmiş verinin ve düşük boyutlu yerleştirmelerin beklenen yerlere ve beklenen tanımlayıcı sırasıyla aktarıldığını denetleyin.

7.3 Herkese açık veriler

Yayımlanmış matrisler GEO ek dosyalarından, 10x Genomics veri sayfasından, CELLxGENE Discover veya Human Cell Atlas (HCA) veri portalından sağlanabilir. Biçimler değişir: mtx üçlüleri, loom, `.h5ad` veya `.rds`. Ham sayım varsaymadan önce eşlik eden README ve veri açıklamasını okuyun.

Dikkat edilmesi gerekenler

Paylaşılan nesneler önceden normalleştirilmiş, logaritmik dönüşümden geçmiş veya hücre tipleriyle etiketlenmiş olabilir. İş akışını yeniden çalıştırmadan önce AnnData içindeki `adata.X` ve `adata.raw` alanlarını, Seurat içindeki `so[["RNA"]]$counts` ile `so[["RNA"]]$data` ayrımını inceleyin. Normalleştirilmiş bir gösterimi yeniden normalleştirmeyin. Küratörce değerlendirilmiş etiketler yeni soruya uygunsa kullanılabilir; yeniden kümeleme bilimsel olarak gerekçeliyse özgün etiketleri koruyun ve yeni analizi açıklayın.

8 Kalite kontrolü ve hücre çağırma

Hücre içeren damlacıkları belirleme, ortam RNA'sı düzeltmesi, hücre kalitesine göre filtreleme ve doublet saptama, kalite değerlendirmesinin farklı sorularını ele alır. Bu ayrım önemlidir: Hücre çağrısı almış bir barkod mutlaka yüksek kaliteli bir singlet (tek hücre olarak sınıflandırılan damlacık) değildir. Bu nedenle örnekleri birleştirmeden önce her yakalama işleminin sonuçlarını ayrı inceleyin [1, 2].

8.1 Ham damlacıklardan hücre çağırma İsteğe bağlı

Filtrelenmiş Cell Ranger veya STARsolo matrisi zaten çağrılmış hücreler içerir. Böyle bir girdide ek hücre çağırmayı atlayın. Ham matris için barkodların sayım sıraları dağılımını inceleyin ve aday damlacıkları EmptyDrops ile tahmin edilen ortam profiline karşı test edin; yöntem tek bir sayım eşiğinden fazlasını kullanır [10].

R

```
library(DropletUtils)
sce <- read10xCounts("sample1/outs/raw_feature_bc_matrix", col.names = TRUE)
br <- barcodeRanks(counts(sce))
set.seed(42)
e <- emptyDrops(counts(sce))
keep <- which(e$FDR < = 0.001)
sce <- sce[, keep]
```

Girdiler ve çıktılar

Girdi: Ortam RNA'sı tahmininde kullanılan damlacıklar da dâhil, filtrelenmemiş sayımlar. Çıktı: Çağrılmış hücre alt kümesi ve tanılar. Gösterilen yanlış keşif oranı (FDR) eşiği 0,001'dir; hücre çağrılarını kabul etmeden önce barkod sıralaması dağılımını, ortam modelini ve simülasyon çözünürlüğünü inceleyin. USA içe aktarma dalında kullanılacak korunmuş barkodları açıkça dışa aktarın.

Dikkat edilmesi gerekenler

Test dışı bırakılan çok düşük sayımlı damlacıklarda NA istatistikleri beklenir; bunlar genellikle yetersiz Monte Carlo örneklemesini göstermez. Karar eşiğine yakın test edilmiş damlacıklarda `Limited` bayrağını inceleyin; simülasyon çözünürlüğü sınırlayıcıysa kurulu sürüm için belgelenmiş argümanla iterasyon sayısını artırın. Rastgelelik tohumu belirleyin. Filtrelenmiş girdi, bu testin gerektirdiği arka plan popülasyonunu sağlayamaz.

8.2 Ortam RNA'sı değerlendirmesi İsteğe bağlı

Süspansiyona salınan RNA, hücre içeren damlacıkların sayımlarına katkıda bulunabilir. SoupX, ham damlacık arka planı ve kümelerle ilişkili ifadeyi kullanarak bulaşmayı tahmin eder; DecontX olasılıksal bir karışım yaklaşımı kullanır [11, 12]. Başka bir popülasyonun belirteçlerindeki düşük ifade veya koşulla ilişkili sinyalin bulaşmayla açıklanıp açıklanamayacağını değerlendirin.

R

```
# Demonstration: define these matrices and preliminary cluster labels first.
# Matrices are genes x barcodes from the same capture and annotation.
library(SoupX)
sc <- SoupChannel(tod = raw_matrix, toc = filtered_matrix)
sc <- setClusters(sc, preliminary_clusters)
sc <- autoEstCont(sc)
corrected <- adjustCounts(sc, roundToInt = TRUE)
```

Ön kümeler bulaşma tahminini destekler; son biyolojik etiketler değildir. Özgün sayımları ve tanıları tahmin edilmiş düzeltilmiş sayımlardan ayrı koruyun. Düzeltme veri modelini değiştirir; her ham sayım katmanını sessizce değiştirmek yerine seçilen gösterimi ve sonraki sayım temelli çıkarım için uygunluğunu doğrulayın. Temel kod yolu ortam RNA'sı düzeltmesini otomatik uygulamaz.

Uygulama notu

Ortam profillerini yakalama işlemi düzeyinde tahmin edin. İlgisiz yakalama işlemlerinden gelen verileri düzeltmeden önce havuzlamak arka planları karıştırır. Çoklanmış tasarımlarda yakalama işlemi ve donör farklı alanlardır; hangi hücrelerin aynı ortam havuzunu paylaştığını analizden önce belirleyin.

8.3 Hücre başına ölçütleri inceleyip filtre seçme

Toplam sayımı, saptanan gen sayısını, mitokondriyal oranı ve diğer ilgili özellik özetlerini hesaplayın. Düşük sayımlar, yüksek mitokondriyal oranlar veya olağan dışı yüksek özellik sayıları sorunlu hücreleri gösterebilir; ancak hiçbiri evrensel bir biyolojik tanı değildir. Doku, hazırlama yöntemi, derinlik ve hücre tipi bu dağılımları etkiler [1, 15].

R

```r
so <- CreateSeuratObject(m, min.cells = 0, min.features = 0)
so[["percent.mt"]] <- PercentageFeatureSet(so, pattern = "^MT-")
so[["percent.ribo"]] <- PercentageFeatureSet(so, pattern = "^RP[SL]")
# Illustrative human-cell thresholds; inspect distributions first.
so <- subset(so, subset = nFeature_RNA > 200 & nFeature_RNA < 6000 & percent.mt < 15)
```

Bu sembol desenleri, Ensembl tanımlayıcıları veya her tür için değil, insan gen sembolleri için geçerlidir. Kod, kararlı tanımlayıcı/sembol tablosunu korur ve mitokondri deseni hiçbir özellikle eşleşmezse durur. Tanımlayıcı uyumsuzluğundan kaynaklanan sıfır mitokondriyal oran, çok yüksek kalite olarak yorumlanmamalıdır.

R

```r
# Bioconductor alternative; read10xCounts stores symbols in rowData.
mito <- grep("^MT-", rowData(sce)$Symbol)
qc <- scuttle::perCellQCMetrics(sce, subsets = list(Mito = mito))
discard <- scuttle::perCellQCFilters(qc, sub.fields = "subsets_Mito_percent")
sce <- sce[, !discard$discard]
```

PYTHON

```python
adata.var["mt"] = adata.var_names.str.startswith("MT-")
sc.pp.calculate_qc_metrics(adata, qc_vars=["mt"], percent_top=None, inplace=True)
adata = adata[(adata.obs.pct_counts_mt < 15) &
              (adata.obs.n_genes_by_counts > 200) &
              (adata.obs.n_genes_by_counts < 6000)].copy()
```

Temel parametreler

Ölçütleri her örnek ve ilgili hücre popülasyonu için inceleyin; medyan mutlak sapmaya (MAD) dayalı aykırı değerler olağan dışı gözlemleri belirlemeye yardımcı olabilir, ancak gerçek nadir tipleri otomatik silmemelidir. Korunan/elenen hücre sayılarını ve nedenlerini kaydedin. Kod önce örnek başına QC ölçütlerini yazar ve `SC_QC_REVIEWED=yes` olana kadar durur; eşik değişkenleri tüm dokular için öneri değil, açık bir başlangıç yapılandırmasıdır.

8.4 Doublet saptama

Doublet gözlemleri farklı ifade profillerini birleştirerek yanıltıcı karma popülasyonlar oluşturabilir. scDblFinder ve Scrublet yapay doublet verileri üretir ve gözlenen profillerle karşılaştırır [16, 49]. Bu yöntemler, bileşen hücreleri birbirine çok benzeyen doublet gözlemlerinde daha az bilgilendiricidir; singlet tahmini, hiç doublet kalmadığını garanti etmez. DoubletFinder, R için başka bir alternatiftir [17].

R

```r
library(scDblFinder)
# sce$capture must identify independent droplet captures.
set.seed(42)
sce <- scDblFinder(sce, samples = "capture",
                   BPPARAM = BiocParallel::SerialParam(RNGseed = 42))
sce <- sce[, sce$scDblFinder.class == "singlet"]
```

PYTHON

```
import scrublet as scr
scrub = scr.Scrublet(adata.X, random_state=42)  # raw counts, one capture
scores, predicted = scrub.scrub_doublets()
adata.obs["doublet_score"] = scores
adata = adata[~predicted].copy()
```

Kod, her örnekte bir yakalama işlemi varsayımı altında her yakalama işleminden gelen veriyi birleştirmeden önce ayrı işler. Birleştirilmiş SingleCellExperiment nesnesi de `samples` argümanı yakalama işlemi değerlerini açıkça tanımlıyorsa doğru işlenebilir. Puan dağılımlarını ve eşik davranışını inceleyin; beklenen oranı belirlerken ölçüme özgü yükleme/geri kazanım rehberini kullanın. Bin hücre başına doublet oranının tüm kimyalarda geçerli tek bir değeri yoktur.

Dikkat edilmesi gerekenler

Son kümeleri yorumlamadan veya koşulları test etmeden önce olası doublet gözlemlerini çıkarın ya da araştırın. Yalnızca karışık belirteçler veya UMAP üzerinde bir köprü kaldığı için ara bir biyolojik durum ilan etmeyin. Tersine, hesaplamalı bir doublet puanı da tek başına nadir bir popülasyonun artefakt olduğunu göstermez; kanıtını ve bağımsız örnekler arasındaki tutarlılığını inceleyin.

9 Normalleştirme ve özellik seçimi

Normalleştirme ve özellik seçimi, veriyi hücrelerin keşif amaçlı karşılaştırılmasına ve düşük boyutlu gösterimin kurulmasına hazırlar. Normalleştirme, farklı dizileme derinliklerindeki hücreleri karşılaştırmayı kolaylaştırır; özellik seçimi ise gösterimde kullanılacak genleri belirler. Bu işlemler tüm teknik etkileri ortadan kaldırmaz; normalleştirilmiş ifade de mutlak RNA ölçeğini korumaz. Hücre tipi anotasyonu ve örnek düzeyindeki çıkarım için genlerin tamamını ve özgün UMI sayımlarını koruyun [1, 2].

Girdiler ve çıktılar

Girdi: Her örnek için kalite kontrolü uygulanmış singlet verileri; örnek, donör, yakalama işlemi ve koşul bilgileri. Çıktı: Korunmuş ham sayımlar, normalleştirilmiş ifade gösterimi, yüksek değişkenlik gösteren genlerin (HVG) maskesi ve temel bileşen analizi (PCA) girdisi. R aşaması `results/objects/qc.rds` dosyasını okur ve `results/objects/pca.rds` dosyasını yazar; Python keşif yolu, sayımları ve normalleştirilmiş gösterimi son `.h5ad` dosyasında birlikte saklar.

9.1 Bir normalleştirme yolu seçin

Genel logaritmik normalleştirme, her hücrenin sayımlarını o hücrenin toplamına böler, seçilen ölçek katsayısıyla çarpar ve `log1p` uygular; örnekte katsayı 10.000'dir. Bu pratik keşif yaklaşımı, göreli kütüphane toplamlarının derinlik farkı için yararlı bir düzeltme sunduğunu varsayar. Hücre bileşimindeki büyük değişimler veya toplam RNA miktarındaki farklılıklar bu varsayımı güçleştirebilir.

SCTransform, düzenlileştirilmiş negatif binom modeli kurar ve varyansı kararlılaştırmak için Pearson artıkları kullanır [23]. Seurat'ın olağan normalleştirme, özellik seçimi ve ölçekleme çağrılarının yerini alır; girdi olarak kullanılan RNA sayımları ayrıca korunmalıdır. scran, büyüklük katsayıları tahmini için hücreleri havuzlar, bu katsayıları ayrıştırma ile hücrelere geri dağıtır ve logaritmik normalleştirilmiş ifade değerleri oluşturur [13, 14]. Bunlar art arda uygulanacak üç düzeltme değil, alternatif iş akışlarıdır.

R

```r
# Minimal Seurat route; counts remain in the RNA assay.
so <- NormalizeData(so, normalization.method = "LogNormalize", scale.factor = 10000)
so <- FindVariableFeatures(so, selection.method = "vst", nfeatures = 2000)
so <- ScaleData(so, features = VariableFeatures(so))
```

Seurat v5, eşlik eden QC aşamasında nesneler birleştirildiğinde örneklere ait ayrı veri katmanlarını korur. Normalleştirme ve değişken özellik seçimi bu katmanlarda doğrudan çalışabilir; zaten bölünmüş bir ölçüm veri yapısı yeniden bölünmemelidir. Belirteç gen testi veya tek bir sayım matrisi çıkarımı öncesinde RNA katmanlarını birleştirin. Tek örnekli nesne, katman bölme veya bütünleştirme gerektirmez.

R

```r
# Alternative Seurat route, starting again from QC'd RNA counts.
so <- SCTransform(so, assay = "RNA", verbose = FALSE)
# Prepare RNA data separately for the later RNA-based examples.
so <- NormalizeData(so, assay = "RNA", normalization.method = "LogNormalize",
                    scale.factor = 10000)
DefaultAssay(so) <- "SCT"  # PCA uses SCT; original RNA counts remain unchanged.
```

Bu yolda PCA, SCT içindeki değişken özellikleri ve ölçeklenmiş artıkları kullanır. Ayrı `NormalizeData(assay="RNA")` çağrısı, sonraki RNA belirteç testleri, SingleR girdisi ve ifade grafikleri için logaritmik normalleştirilmiş RNA verisi oluşturur; SCT'ye yeniden normalleştirme uygulamaz. Her iki işlem de pseudobulk için kullanılan özgün RNA sayımlarını korur. Yalnızca varsayılan ölçüm veri yapısını değiştirmek, eksik bir RNA veri katmanı oluşturmaz.

R

```r
# Alternative Bioconductor route, starting from counts(sce).
library(scran)
library(scater)
set.seed(42)
clusters <- quickCluster(sce)
sce <- computeSumFactors(sce, clusters = clusters)
sce <- logNormCounts(sce)
```

PYTHON

```python
import scanpy as sc

# Save before any operation that overwrites X; keep all genes.
adata.layers["counts"] = adata.X.copy()
sc.pp.normalize_total(adata, target_sum=1e4)
sc.pp.log1p(adata)
```

Dikkat edilmesi gerekenler

Bir sayım katmanı, logaritmik normalleştirilmiş ifade veya Pearson artıkları yerine sonlu, negatif olmayan tam sayı UMI sayımları içerir. AnnData içindeki `.raw`, analistin seçtiği bir gösterimin kaydıdır; adı, ham sayımları içerdiğini kanıtlamaz. Dönüştürülmüş değerleri yuvarlamak özgün sayımları geri getirmez. scVI ve pseudobulk için korunmuş sayım katmanını, betimleyici belirteç grafiklerinde ise uygun normalleştirilmiş gösterimi kullanın.

9.2 Yüksek değişkenlik gösteren genlerin seçimi

Yaklaşık 2.000 HVG, keşif amaçlı PCA için başlangıç seçeneğidir; biyolojik bir sabit değildir. Daha çok gen, daha zayıf hücre durumu sinyallerini koruyabilir; teknik veya bilgi taşımayan değişkenliği de artırabilir. Çok az gen seçmek ise nadir bir popülasyonu ayırt eden özellikleri dışarıda bırakabilir. Seçilen genler PCA'yı, tutulan bileşenler de uzaklıkları ve komşuları belirlediğinden özellik seçimi kümelere ve UMAP'a kadar yansıyabilir. Seçilen genlerin ve PCA'dan elde edilen temel bileşen (PC) değerlerinin, ilgilenilen hücre durumlarını mı, dizileme derinliğini mi yoksa başka bir istenmeyen etkiyi mi yansıttığını inceleyin. Seurat'ın `vst` seçimi ortalama–varyans ilişkisini modeller. Scanpy içindeki `seurat_v3` seçeneği ham sayımlar ve ek `scikit-misc` bağımlılığı gerektirir; girdisi, logaritmik verilerle çalışan `seurat` seçeneğinden farklıdır.

R

```r
# Already included in the minimal Seurat route above.
top10 <- head(VariableFeatures(so), 10)

# scran alternative, using log-normalised expression in sce.
dec <- modelGeneVar(sce)
hvgs <- getTopHVGs(dec, n = 2000)
```

PYTHON

```python
sc.pp.highly_variable_genes(
    adata, n_top_genes=2000, flavor="seurat_v3",
    layer="counts", batch_key="sample",
)
# Preserve non-HVGs; PCA uses a feature mask instead of slicing them away.
sc.pp.pca(adata, n_comps=30, mask_var="highly_variable", random_state=42)
```

Python örneği HVG'leri örnekler içinde seçer ve sıralamalarını birleştirir. Böylece yalnızca örneğe özgü değişkenliğin yönlendirdiği seçim azaltılabilir; bu işlem, örnekler arasındaki tüm farkların teknik olduğunu göstermez. Seurat ve Scanpy uygulamalarının aynı genleri seçmesi veya sayısal olarak özdeş düşük boyutlu yerleştirmeler üretmesi beklenmemelidir.

2025 tarihli bir karşılaştırma, on veri kümesinde 24 özellik seçimi yöntemini değerlendirmiş; varyansı kararlılaştıran HVG yöntemlerinin bütünleştirme ve sorgu eşleme görevlerinde etkili olduğunu göstermiştir. Yaklaşık 2.000 özelliği destekleyen bu sonuç göreve bağlıdır; diğer genleri atmayı gerektirmez veya evrensel bir optimum tanımlamaz. Nadir gruplar, hücre durumları ya da bütünleştirme görevi farklıysa özellik seçimini yeniden değerlendirin [50].

9.3 Ölçekleme ve istenmeyen değişkenlik

Seurat'ın logaritmik normalleştirme yolu, PCA öncesinde seçilen özellikleri merkezler ve ölçekler. SCTransform kendi artık gösterimini sağlar. Python kodu, ek bir varyans ölçeklemesi yapmadan logaritmik normalleştirilmiş HVG'ler üzerinde PCA uygular; bu, açıkça belirtilmiş alternatif bir gösterimdir ve PC'leri Seurat yolunun bire bir karşılığı değildir. Python'da ölçekleme seçilirse uygun bir çalışma gösterimine uygulayın; sayımları ve logaritmik ifade değerlerini ayrı tutun.

R

```r
# Optional regression in a log-normalised Seurat analysis.
# so <- ScaleData(so, features = VariableFeatures(so),
#                 vars.to.regress = "percent.mt")
```

Temel parametreler

Normalleştirme yöntemini, ölçek katsayısını, özellik seçimi yöntemini, HVG sayısını ve kullanılan teknik parti değişkenlerini veya regresyon ortak değişkenlerini kaydedin. Eşlik eden kod 2.000 HVG ve en çok 30 PC ister; matris daha küçükse PC sayısını azaltır. Bütünleştirme seçmeden önce düzeltilmemiş gösterimi karşılaştırın (Bölüm 10).

Dikkat edilmesi gerekenler

Bir ortak değişkenin etkisini yalnızca bu sinyalin kaldırılması bilimsel soruya hizmet ediyorsa regresyonla giderin. Mitokondriyal ifade ve hücre döngüsü etkinliği, kalite veya teknik değişkenliğin yanı sıra biyolojiyi de yansıtabilir. Rutin regresyon; stres altındaki, çoğalan veya farklılaşan popülasyonlardaki gerçek farkları silebilir. Regresyonun gösterimi iyileştirdiğini varsaymak yerine ortaya çıkan yapıyı yeniden değerlendirin.

10 Boyut indirgeme ve bütünleştirme

PCA, seçilen ifade gösterimindeki değişkenliği bileşenlerle özetler; bu bileşenler keşif amaçlı komşuluk çizgesinin kurulmasında kullanılır. Bileşenlerdeki yapı biyolojiyi, teknik değişkenliği veya ikisini birlikte yansıtabilir. Bu yüzden son ortak gösterimi oluşturmadan önce bütünleştirme gerekip gerekmediğini ayrıca değerlendirin [1, 28].

10.1 PCA

Açıklanan varyansı ve önemli temel bileşenlere katkıda bulunan genleri; örnek, kalite ve biyolojik etiketlerle birlikte inceleyin. 20–50 bileşen gibi bir aralık, otomatik kesme değeri değil keşif başlangıcıdır. Görünür bir dirsek noktası, tek başına hangi değişkenliğin biyolojik olduğunu göstermez.

R

```
set.seed(42)
so <- RunPCA(so, features = VariableFeatures(so), npcs = 30, seed.use = 42)
ElbowPlot(so, ndims = 30)
reduction <- "pca"
dims <- seq_len(ncol(Embeddings(so, reduction)))
```

PYTHON

```
sc.pp.pca(adata, n_comps=30, mask_var="highly_variable", random_state=42)
sc.pl.pca_variance_ratio(adata, n_pcs=30, log=True)
representation = "X_pca"
```

Metin içindeki örnekler 30 PC ister; bunun için yeterli hücre ve seçilmiş özellik bulunmalıdır. Eşlik eden kod ise 30, değişken özellik sayısının bir eksiği ve hücre sayısının bir eksiği arasından en küçüğünü alır; sonuç iki PC'nin altındaysa durur. Ek PC'ler ince hücre durumu farklarını koruyabileceği gibi derinlik veya kalite değişkenliğini de ekleyebilir; PC'leri dışlamak her iki tür sinyali de silebilir. Daha çok PC'nin analizi iyileştirdiğini varsaymak yerine gen yüklerini ve çizge ile etiketlerin bu seçime duyarlılığını inceleyin. Seurat'ın ölçeklenmiş özelliklerden, Python kodunun ise ölçeklenmemiş logaritmik ifadeden ürettiği PC'ler farklı gösterimlerdir; sayısal eşitlik doğrulama hedefi değildir.

10.2 Bütünleştirme gerekip gerekmediğine karar verme

Önce düzeltilmemiş bir gösterim oluşturun; örnekler, bilinen teknik değişkenler ve olası hücre tipleri açısından inceleyin. Örneklerin ayrılması teknik etkiden, gerçek koşul yanıtından, farklı hücre popülasyonlarından veya yetersiz QC'den kaynaklanabilir. UMAP görünümü tek başına nedeni belirleyemez. Teknik parti ile koşulun birbirinden ayrılamayacak biçimde tamamen örtüşmesi bütünleştirme ile giderilemez.

Uygulama notu

Kod varsayılan olarak düzeltilmemiş PCA kullanır. Tasarım ve düzeltilmemiş yapı değerlendirildikten sonra `SC_INTEGRATION_KEY`, mevcut bir üst veri alanıyla Harmony isteyebilir. Özgün PCA'yı ve tanılarını koruyun. `sample` seçimi, keşif amaçlı eşleme için örnekle ilişkili değişkenliği hizalar; kaldırılan tüm değişkenliğin teknik olduğunu göstermez.

10.3 Bütünleştirme alternatifleri

Harmony, PCA gösterimini iteratif olarak düzeltir [24]. Seurat'ın çapa temelli CCA/RPCA yöntemleri veri kümeleri arasında karşılıklar belirler [19]; fastMNN, düşük boyutlu uzayda karşılıklı en yakın komşuları eşler [25]. scVI, teknik parti ortak değişkenleri içeren bir üretici sayım modeli kurar ve örtük gösterim sağlar [26, 27]. Varsayımları, hesaplama gereksinimleri ve biyolojik değişkenliği koruma davranışları farklıdır. Karşılaştırmalı değerlendirme sonuçları, performansın veriye ve değerlendirme hedefine bağlı olduğunu gösterir [28]; hiçbir yöntem her durumda en iyi değildir.

R

```r
# Optional Harmony branch after normalization and PCA.
# Confirm that this field is the intended grouping variable.
so <- JoinLayers(so, assay = "RNA")
set.seed(42)
so <- harmony::RunHarmony(so, group.by.vars = "sample", reduction.use = "pca")
reduction <- "harmony"
# Alternative: IntegrateLayers on appropriately split Seurat v5 layers.
```

PYTHON

```python
# Optional scVI demonstration; requires preserved integer counts.
import scvi
scvi.settings.seed = 42
model_data = adata[:, adata.var.highly_variable].copy()
scvi.model.SCVI.setup_anndata(model_data, batch_key="sample", layer="counts")
model = scvi.model.SCVI(model_data)
model.train()
assert model_data.obs_names.equals(adata.obs_names)
adata.obsm["X_scVI"] = model.get_latent_representation()
representation = "X_scVI"
```

Temel parametreler

Hem istenmeyen teknik yapının giderilmesini hem ilgili biyolojik farkların korunmasını değerlendirin. Güçlü teknik parti karışımı tek başına yeterli değildir; aşırı düzeltme de gerçekten farklı durumları karıştırabilir. Seçilen yöntemi, gruplama değişkenlerini, özellik kümesini, parametreleri ve düzeltilmemiş veriyle karşılaştırmayı kaydedin. Örnek düzeyindeki ifade çıkarımı için özgün RNA sayımlarını koruyun (Bölüm 12).

Hücre tiplerinin birbirinden ayrılmasının yanında her tipin içindeki yapının korunmasını da değerlendirin. Akciğer ve meme atlaslarını kullanan bir karşılaştırma, geniş etiketlere dayalı bütünleştirme puanlarının tip içi yapının kaybını gözden kaçırabildiğini göstermiştir [51]. Düzeltilmiş ve düzeltilmemiş gösterimleri beklenen durumlar ve örnek düzeyindeki ifade kanıtlarıyla karşılaştırın; yalnızca UMAP'ta örneklerin iyi karışmış görünmesine göre seçim yapmayın. 2026 tarihli bir perspektif de aşırı ve yetersiz düzeltmenin süren sınırlılıklar olduğunu belirtmektedir [52].

10.4 Çizge kurma ve görselleştirme

UMAP, seçilen PCA veya örtük gösterimdeki yerel yapıyı görselleştirir [29]; t-SNE başka bir gösterim seçeneğidir [30]. Amaçlanan çizge temelli kümeleme için aynı gösterimi kullanın ve her şeklin hangi gösterimi yansıttığını belirtin.

R

```
dims <- seq_len(ncol(Embeddings(so, reduction)))
set.seed(42)
so <- FindNeighbors(so, reduction = reduction, dims = dims)
set.seed(42)
so <- RunUMAP(so, reduction = reduction, dims = dims, seed.use = 42)
DimPlot(so, reduction = "umap", group.by = "sample")
```

PYTHON

```
sc.pp.neighbors(adata, use_rep=representation, random_state=42)
sc.tl.umap(adata, random_state=42)
sc.pl.umap(adata, color="sample")
```

Yukarıdaki Seurat çağrıları aynı PC'leri kullanır, ancak ayrı komşuluklar kurar: `FindNeighbors`, kümeleme çizgesi için `k.param`; `RunUMAP` ise görselleştirme komşuluğu için `n.neighbors`, noktaların ne kadar sıkı yerleşebileceği için `min.dist` kullanır. `min.dist` arttığında görünüm genellikle yayılır; önceden hesaplanmış kümeler değişmez. Aynı gösterim, aynı çizge anlamına gelmez. Scanpy UMAP'ı ise `sc.pp.neighbors` tarafından saklanan çizgeyi kullanır; yine de görünümün genel uzaklıkları koruması beklenmez.

Dikkat edilmesi gerekenler

UMAP'taki genel uzaklıklar, yoğunluk, küme alanı ve görünür köprüler gösterim algoritmasıyla bozulabilir. UMAP koordinatlarını kümelemeyin veya geometrilerini nicel bir ifade etkisi saymayın. Uygun yüksek boyutlu çizgede kümeleme yapın; ifadeyi ise gerekli tekrar yapısını gözeten modelle gen düzeyindeki veriler üzerinde test edin. Platformlar arasında bit düzeyinde özdeş yerleştirmeler vaat etmeden rastgelelik tohumu ve sürümleri kaydedin.

11 Kümeleme ve hücre tipi anotasyonu

Kümeleme, seçilen ifade gösterimini gruplara ayırır. Hücre tipi anotasyonu, bu grupları belirteçler, referanslar ve biyolojik bağlamla yorumlama aşamasıdır; küme numarası ya da otomatik etiket tek başına hücre kimliğini ortaya koymaz. Analize Bölüm 10 içinde seçilen düzeltilmemiş veya bilinçli olarak bütünleştirilmiş gösterimden başlayın [1, 2].

11.1 Çizge temelli kümeleme

Seurat en yakın komşu çizgesi kurar; çoğunlukla bu çizgeden, paylaşılan komşuları ağırlıklandıran bir paylaşılan en yakın komşu çizgesi (SNN) oluşturur; Scanpy ağırlıklı bir komşuluk çizgesi oluşturur. Topluluk saptama, iki boyutlu UMAP yerine bu çizgelerde çalışır. Leiden, Louvain bölmelerindeki bağlantılılık sorunlarını ele alır [31]; başka bir algoritma seçilmezse Seurat varsayılan olarak Louvain kullanır [32].

R

```
set.seed(42)
so <- FindNeighbors(so, reduction = reduction, dims = dims)
set.seed(42)
so <- FindClusters(so, resolution = 0.5, algorithm = 4, random.seed = 42)
```

PYTHON

```
sc.pp.neighbors(adata, use_rep=representation, random_state=42)
sc.tl.leiden(adata, resolution=0.5, key_added="leiden", random_state=42,
             flavor="igraph", directed=False, n_iterations=2)
```

Seçilen Seurat sürümünün Leiden için kullandığı hesaplama arka ucunu ve bağımlılıklarını denetleyin; eski sürümler reticulate üzerinden Python kullanabilirken güncel arayüzler R arka uçları da sunar. Kod Leiden'i açıkça ister. Rastgelelik tohumu ve çözünürlük değişmese bile paket değişikliği çizgeyi veya bölmeyi değiştirebilir.

Temel parametreler

Komşuluk büyüklüğü çizgenin ölçeğini değiştirir: Küçük komşuluklar yerel yapıyı öne çıkarır ve parçalayabilir; büyük komşuluklar daha geniş bölgeleri bağlar ve ince ayrımları birleştirebilir. Seurat'ta `k.param`, Scanpy'da `n_neighbors` kullanılır; duyarlılığı değerlendirirken her birini açıkça kaydedin. Çözünürlük bölmenin ayrıntı düzeyini belirler; sabit bir çizgede artırılması genellikle daha ince bölmeleri destekler. Ancak 0,5 bir öğretim başlangıç değeridir; farklı yazılım, çizge veya veri kümelerinde eşdeğer bölme sağlamaz. Uygun bir aralığı araştırın, kararlılığı ve belirteç uyumunu inceleyin ve aday popülasyonların bağımsız örneklerde yeniden görülüp görülmediğini değerlendirin. Daha çok küme her zaman daha çok biyolojik bilgi sağlamaz. Gösterimi, komşuları, çizge ayarlarını, kümeleme algoritmasını ve tohumu kaydedin; yalnızca beklenen hücre tipi sayısını elde etmek için çözünürlük seçmeyin.

11.2 Betimleyici belirteç genler

Bir grubu diğerlerinin tümüyle karşılaştıran test, her kümede zenginleşen genleri sıralar. Seurat ve Scanpy, normalleştirilmiş gen ifadesi üzerinde Wilcoxon sıra toplamı testi kullanabilir; seçilen Seurat sürümü presto kullanıyorsa hesaplama hızlanır [33]. Bunlar donör düzeyinde koşul testi değil, betimleyici etiketleme araçlarıdır.

R

```
so <- JoinLayers(so, assay = "RNA")
markers <- FindAllMarkers(so, assay = "RNA", only.pos = TRUE,
                          min.pct = 0.25, logfc.threshold = 0.25, test.use = "wilcox")
```

SCT alternatifi de dâhil, RNA veri yapısı Bölüm 9 içinde hazırlanmış logaritmik normalleştirilmiş değerleri içermelidir; katmanları birleştirmek bu değerleri oluşturmaz. Burada `min.pct=0.25`, karşılaştırılan iki gruptan en az birindeki hücrelerin en az %25'inde saptanan genleri tutar; `logfc.threshold=0.25` ise ortalama logaritmik kat değişimine göre ön eleme yapar. Eşiklerden birini yükseltmek adayları ve hesaplama yükünü azaltır, ancak seyrek veya daha zayıf belirteçleri dışlayabilir. Bunlar donör düzeyinde etki eşikleri değil, betimleyici belirteçler için ön filtrelerdir; `only.pos=TRUE` yalnızca pozitif belirteçleri raporlar.

PYTHON

```
sc.tl.rank_genes_groups(adata, groupby="leiden", method="wilcoxon", use_raw=False)
markers = sc.get.rank_genes_groups_df(adata, group=None)
```

R

```
# Bioconductor alternative, after assigning cluster labels to sce.
markers <- scran::findMarkers(sce, groups = colLabels(sce), test.type = "wilcox")
```

Dikkat edilmesi gerekenler

Aynı veri hem kümelerin tanımlanmasına hem belirteçlerinin bulunmasına katkıda bulunmuştur. Standart belirteç p-değerleri bu seçimi hesaba katmaz; donör bağımlılığını da her zaman dikkate almaz. Etki büyüklükleri, saptanma sıklıkları, bilinen belirteç panelleri ve bağımsız örneklerle etiketlemeyi değerlendirin; küme belirteci tablosunu koşullar arasında diferansiyel gen ifadesi olarak yeniden yorumlamayın (Bölüm 12).

11.3 Referans temelli etiketleme

Aşağıdaki SingleR dönüşümü SCT artıklarını değil, RNA sayımlarını ve hazırlanmış logaritmik normalleştirilmiş RNA verisini kullanır. SingleR, sorgu ifadesini celldex gibi kaynakların sağladığı etiketli referans profilleriyle karşılaştırır [34]. Azimuth, Seurat ekosisteminde referansa eşleme sağlar [35]; CellTypist, bağışıklık ve doku modelleri dâhil eğitilmiş sınıflandırıcılar kullanır [36]. Tür, doku, hazırlama yöntemi ve beklenen popülasyonlarla uyumlu bir referans seçin; belirsiz veya atanmamış tahminleri inceleyin.

R

```r
library(SingleR)
library(celldex)
ref <- HumanPrimaryCellAtlasData()
pred <- SingleR(test = as.SingleCellExperiment(so, assay = "RNA"),
                ref = ref, labels = ref$label.main)
so$SingleR <- pred$labels[match(colnames(so), rownames(pred))]
so$SingleR_pruned <- pred$pruned.labels[match(colnames(so), rownames(pred))]
```

PYTHON

```python
# Optional demonstration: obtain a suitable model first.
import celltypist
model = celltypist.models.Model.load(model="Immune_All_Low.pkl")
pred = celltypist.annotate(adata, model=model, majority_voting=True)
adata.obs["celltypist_label"] = pred.predicted_labels["majority_voting"]
```

CellTypist örneği, modelin belgelenmiş normalleştirilmiş girdisini ve gen tanımlayıcılarını bekler; rastgele ölçeklenmiş herhangi bir matriste çalışmaz. Bir referans, eğitim etiketlerinde bulunmayan popülasyonları güvenilir biçimde atayamaz. Makul görünen etiketler bu nedenle otomatik kabul yerine belirteç ve bağlam denetimi gerektirir.

11.4 İncelenmiş etiketleri doğrulayıp saklama

Kanonik pozitif ve negatif belirteçleri, saptanma oranlarını ve örnek başına temsili inceleyin. Bir küme karışık popülasyon, düşük kaliteli hücreler veya belirsiz etiketler içerebilir; kanıt daha ayrıntılı kimliği desteklemiyorsa atanmamış kategorisini koruyun. Kod, öneri niteliğinde `config/cluster_labels.tsv` yazar, inceleme için durur ve düzenlenmiş eşlemeyi yalnızca `SC_LABELS_REVIEWED=yes` ile uygular.

R

```r
# Example only: replace with a map covering all observed clusters exactly.
new_ids <- c("0" = "CD4 T", "1" = "CD14 Mono", "2" = "B", "3" = "NK")
stopifnot(setequal(names(new_ids), levels(so)))
so$celltype <- unname(new_ids[as.character(Idents(so))])
Idents(so) <- "celltype"
DotPlot(so, assay = "RNA", features = c("CD3D", "CD4", "CD14", "MS4A1", "NKG7"))
```

PYTHON

```
new_ids = {"0": "CD4 T", "1": "CD14 Mono", "2": "B", "3": "NK"}
assert set(new_ids) == set(adata.obs["leiden"].astype(str))
adata.obs["celltype"] = adata.obs["leiden"].astype(str).map(new_ids)
sc.pl.dotplot(adata, ["CD3D", "CD4", "CD14", "MS4A1", "NKG7"], groupby="celltype")
```

Şekillerde ve çıkarımda `celltype` üst veri alanını tutarlı kullanın. Küme numaraları tekrar çalıştırmalar arasında değişebilir; incelenmiş eşlemeyi, referans sürümünü ve destekleyici belirteç kanıtını nesneyle birlikte kaydedin. Python kodu kümeleri ve belirteçleri sağlar, ancak bu etiketleme adımı ayrıca uygulanmalıdır.

12 Koşullar arasında diferansiyel gen ifadesi

Bir koşul karşılaştırması, etkinin bağımsız deney birimleri, çoğu zaman donörler, arasında genellenip genellenemeyeceğini sorgular. Aynı donörden gelen hücreler, ek bağımsız donör tekrarları sağlamaz. Çalışma örneğinde her bağımsız donör için bir kütüphane ve bir yakalama işlemi bulunur; eşleştirilmiş donörler, tekrarlı ölçümler, teknik kütüphaneler ve çoklanmış yakalama işlemleri bu ilişkileri temsil eden bir tasarım gerektirir [37, 38].

Soruyu bulk RNA-seq ile ilişkilendirin. Tüm dokuya ait bulk karşılaştırması, hücre gruplarının kendi içindeki ifadeyi ve RNA karışımına katkılarını birlikte yansıtır. Hücre tipi pseudobulk karşılaştırması ise anotasyonla tanımlanmış grubun içindeki göreli ifadeyi; bileşim analizi, yakalanan hücreler içindeki payını inceler. Bu nicelikler birbirinden bağımsız değişebilir. Sonuçların uyuşması gerekmez; tip içindeki bir etki tek başına tüm dokudaki etkiyi açıklamaz [37, 38].

12.1 Pseudobulk: Ham sayımları örnek içinde toplulaştırın

Pseudobulk yaklaşımı, koşul karşılaştırmasını örnek düzeyinde kurar. Gözden geçirilmiş bir hücre tipine ait hücrelerin özgün UMI sayımları her örnek içinde toplanır; böylece DESeq2 veya edgeR için genler×örnekler matrisi oluşur. Bu matrise uygulanan doğru tanımlanmış model, örnekler arası değişkenliği hesaba katabilir. Yine de bağımsız tekrarlar veya hücre tipi anotasyonu yetersizse, etkiler birbirine karışıyorsa ya da model tasarımı uygun değilse toplulaştırma tek başına geçerli çıkarımı garanti etmez [37, 38, 39, 40].

Küçük bir toplulaştırma örneği. Aşağıdaki kurmaca UMI sayımları, incelenmiş tek bir hücre tipinden beş hücredeki iki gene aittir. A1–A3 donör A'dan, B1–B2 donör B'den gelir. Bu örnekte her donör bir örnek sağlar.

Gen	A1	A2	A3	B1	B2	Donör A	Donör B
G1	2	0	4	1	3	6	4
G2	0	3	1	2	5	4	7

G1 için donör sütunları $2+0+4=6$ ve $1+3=4$; G2 için $0+3+1=4$ ve $2+5=7$ olur. İfade matrisinde iki donör sütunu vardır; ayrı hücre sayısı kaydı ise A için üç, B için iki hücre gösterir. Bağımsız donör sayısı ikidir; beş hücre beş tekrar oluşturmaz. Hücre eklemek toplamları ve örnekleme kapsamını değiştirir, donör sayısını artırmaz. Bu küçük tablo yalnızca toplulaştırmayı gösterir; koşul testi için bir tasarım değildir.

Girdiler ve çıktılar

Girdi: Katmanları birleştirilmiş ve dönüştürülmemiş RNA sayımları, gözden geçirilmiş `celltype` etiketleri ve örnek tablosuyla birlikte `annotated.rds`. Çıktı: Bir hücre tipinin diferansiyel ifade sonucu, uyarlanmış DESeq2 nesnesi ve hücre sayısı tanıları. Eşlik eden komut `<celltype> <numerator> <denominator>` alır; incelenmiş nesnede gerçekten bulunan etiketleri seçin.

R

```r
# Run from the analysis root; SC_WORKFLOW points to this companion folder.
library(Seurat)
library(DESeq2)
source(file.path(Sys.getenv("SC_WORKFLOW"), "R/contracts.R"))
samples <- read_samples("config/samples.tsv")
so <- readRDS("results/objects/annotated.rds")
ct <- "T_cells"  # replace with a reviewed label present in this object
sub <- subset(so, cells = colnames(so)[so$celltype == ct])
raw <- LayerData(sub, assay = "RNA", layer = "counts")
pb <- aggregate_counts(raw, sub@meta.data, group = "sample")
meta <- align_samples(pb, samples, "treated", "control")
stopifnot(setequal(colnames(pb), samples$sample))

dds <- DESeqDataSetFromMatrix(as.matrix(pb), meta, design = ~ condition)
dds <- dds[rowSums(counts(dds) > = 10) > = 2, ]
dds <- DESeq(dds)
res <- results(dds, contrast = c("condition", "treated", "control"))
```

Ortak toplulaştırma yardımcısı örnek tanımlayıcılarını aynen korur, üst veri sırasını ve toplam sayımların korunmasını denetler. Örnek üst verisini bileşik sütun adlarından yeniden kurmaktan kaçınır: Seurat'ın `AggregateExpression` işlevi de sayımları toplar, ancak oluşturduğu grup adlarında alt çizgileri tireye çevirebilir. Koşulu örnek adının ön ekinden çıkarmayın. Yukarıdaki yoğun matrise dönüştürme, hücrelerin tamamına değil küçük pseudobulk matrisine uygulanır. Eşlik eden DE tablosu, korunmuş özellik eşlemesinden gelen kararlı `gene_id`, özgün `gene_symbol` ve nesnedeki `row_name` sütunlarını içerir; görüntüleme amaçlı gen sembolleri tek başına kararlı eşleme tanımlayıcıları değildir.

muscat, `aggregateData` ve `pbDS` aracılığıyla birden fazla hücre tipi için alternatif bir iş akışı sağlar [38]. Toplulaştırma arayüzünü değiştirirken aynı deney birimlerini, ham sayım gereksinimini ve tasarım kontrollerini koruyun.

PYTHON

```python
# Demonstration after reviewed labels have been added to the full-gene object.
import os
import sys
from pathlib import Path
import numpy as np
from scipy import sparse
import scanpy as sc
sys.path.insert(0, str(Path(os.environ["SC_WORKFLOW"]) / "py"))
from workflow_contracts import read_samples, align_metadata, check_cell_metadata
samples = read_samples()
check_cell_metadata(adata.obs, samples)

assert "counts" in adata.layers
pb = sc.get.aggregate(
    adata, by=["sample", "celltype"], func="sum", layer="counts",
)
pb_ct = pb[pb.obs["celltype"] == "T_cells"].copy()
pb_ct.X = pb_ct.layers["sum"]
values = pb_ct.X.data if sparse.issparse(pb_ct.X) else np.asarray(pb_ct.X)
assert np.isfinite(values).all() and (values > = 0).all()
assert (values == np.floor(values)).all()
# R count matrices use genes as rows and samples as columns.
counts_table = pb_ct.to_df().T
counts_table.columns = pb_ct.obs["sample"].to_numpy()
counts_table.to_csv("pseudobulk_T_cells_counts.csv")
meta = align_metadata(pb_ct.obs["sample"].tolist(), samples)
assert counts_table.columns.tolist() == meta.index.tolist()
meta.to_csv("pseudobulk_T_cells_metadata.csv", index_label="sample_id")
```

Bu Python bloğu dışa aktarım hazırlar; koşul modeli kurmaz. DESeq2, edgeR veya ayrıca doğrulanmış bir PyDESeq2 iş akışı kullanmadan önce dışa aktarılan örnek sütunları ile üst veri satırlarının bire bir eşleştiğini denetleyin. Eşlik eden Python kodu betimleyici küme belirteçleri aşamasında sona erer; incelenmiş hücre tipi etiketlerini veya koşul düzeyindeki diferansiyel ifadeyi otomatik olarak sağlamaz.

Temel parametreler

Beklenen etki ve heterojenlik için yeterli bağımsız donör planlayın; koşul başına üç veya daha fazla donör pratik bir başlangıç önerisidir, güç hesabı veya garanti değildir. Kod, her koşulda korunmuş en az iki bağımsız örnek ve seçilen hücre tipinin listelenen her örnekte bulunmasını ister. Model öncesinde örnek/hücre tipi başına hücre ve kütüphane büyüklüğünü inceleyin. Örnek sayım filtresi açıklama amaçlıdır. Eşleştirilmiş veya teknik partiler içeren bir tasarım, `~condition` modelinin otomatik yeniden kullanımını değil terimlerinin etkileri ayrı ayrı tahmin edilebilen uygun bir model gerektirir; koşul ile teknik parti tamamen örtüşüyorsa etkileri ayrı tahmin edilemez.

Uygunluk ve tanısal kontroller. Bir örnekte ilgili tipten hiç hücre yakalanmaması, o tipin ifadesinin gözlenmediği anlamına gelir; tümü sıfır bir ifade profili değildir. Sıfır sayımlı kütüphane eklemeyin veya örneği sessizce çıkarmayın. Hücre geri kazanımını, kalite kontrolünü ve anotasyonu araştırın; örnek/tip uygunluğunu istenen DE sonucuna göre seçmeden tanımlayın. Uygun örneklerde pseudobulk kütüphane büyüklüklerini, dönüştürülmüş örnek düzeyi PCA'yı, sayım profillerini ve etkili donörleri inceleyin. Ayrıntılı bir etiket altında bile koşullara göre farklı alt durumlar bulunabilir; DE sonucu, etiket içindeki bu karışımın değişmesini de yansıtır [2, 38].

12.2 Havuzlanmış hücre başına testler koşul çıkarımında neden başarısız olur?

Hücreleri örneklerini dikkate almadan koşula göre havuzlayan Wilcoxon veya t-testi, ilişkili gözlemleri bağımsız sayar. Donör düzeyindeki soruyla ilgili değişkenliği olduğundan küçük tahmin eder ve çok sayıda

yanlış pozitif bulgu üretebilir [37]. Aynı az sayıdaki donörden daha çok hücre eklemek, bağımsız donörlerin yerini tutmaz. Betimleyici küme belirteci testleri farklı bir soruyu yanıtlar; koşul etkisini göstermez (Bölüm 11).

Dikkat edilmesi gerekenler

Örnek yapısını gözeten çıkarım kullanın: Uygun tasarımla pseudobulk veya tekrarlı gözlemleri temsil eden, gerekçelendirilmiş bir karma model. Aynı donörün teknik kütüphaneleri bağımsız biyolojik tekrar olarak sayılamaz. Hiçbir model, koşulla tamamen örtüşen bir teknik etkiden koşul etkisini ayıramaz.

12.3 Diferansiyel bolluk: Göreli bileşim

Bir koşul, hücre tipi içindeki ifade değişmeden de o tipe atanan yakalanmış hücrelerin oranını değiştirebilir. Propeller, örnek başına oranları hesaplar, dönüştürür ve düzenlileştirilmiş varyans tahminli doğrusal model çerçevesi ile grup farklarını test eder [42]. Milo ise bir k-en yakın komşu çizgesi (kNN) üzerindeki örtüşen komşuluklarda hücre sayılarını örnek düzeyindeki tasarım ve normalleştirmeyle test eder [41]. Bunlar bolluk sorusunu farklı çözünürlüklerde ele alan alternatiflerdir; mutlak hücre sayısını sınayan birbirinin yerine geçebilir testler değildir.

Payda örneği. Kurmaca bir örnekte yakalanan 20 X hücresi ve 80 Y hücresi, $20/100 = 20\%$ ve $80/100 = 80\%$ paylarını verir. Başka bir örnekte 40 X ve 80 Y hücresi bulunduğunda paylar $40/120 \approx 33{,}3\%$ ve $80/120 \approx 66{,}7\%$ olur. Yakalanan Y hücresi sayısı değişmediği hâlde payda büyüdüğü için payı düşer. Bunlar gen UMI toplamları değil, hücre sayılarıdır; örnekler mutlak doku bolluğunu veya koşul etkisini göstermez.

R

```
library(speckle)
out <- propeller(clusters = so$celltype, sample = so$sample,
                 group = so$condition, transform = "logit")
```

Dikkat edilmesi gerekenler

Oranların toplamı birdir: Bir popülasyonun artması, diğer popülasyonların mutlak sayıları değişmese bile göreli oranlarını azaltabilir. Dönüşüm bu bağımlılığı ortadan kaldırmaz veya mutlak bolluğu ortaya koymaz. Propeller ve Milo sonuçlarını paydaları ve örnekleme varsayımları çerçevesinde yorumlayın; doku ayrıştırma, yakalama verimi ve eşit olmayan hücre geri kazanımı da gözlenen bileşimi değiştirebilir. Bağımsız örnekler kullanın ve grup testini yorumlamadan önce örnek başına oranları inceleyin.

İsteğe bağlı Bir pseudobulk gen listesi üzerinde işlevsel zenginleşme için clusterProfiler ile aşırı temsil analizi veya sıralı liste yöntemi kullanılabilir [48]. Testte kullanılan arka plan gen kümesini, ilgili hücre tipinin diferansiyel analizine uygun genlerle tanımlayın; tanımlayıcı eşlemelerini koruyun, seçim ve çoklu test kurallarını bildirin. Bu isteğe bağlı adım, temel tek hücre kodlarında uygulanmaz.

13 Yörünge ve RNA velocity İsteğe bağlı

Bu analizler, sürekli veya dinamik bir süreç hakkında ek bir biyolojik soru gerektirir. Tedavi–kontrol karşılaştırmasının otomatik uzantıları değildir. Statik bir hücre tipi koleksiyonu, hücre soy hattı çıkarımı için gereken ara durumları, zaman bilgisini veya örnekleme yapısını içermeyebilir. Önerilen süreci incelerken özgün hücre kimliklerini, donörleri ve koşul etiketlerini koruyun.

13.1 Yörünge çıkarımı ve göreli süreç sıralaması

Yörünge yöntemleri ifade durumları içinde bir yol veya çizge çıkarır ve hücrelere göreli süreç sıralaması (pseudotime) atar. Slingshot, küme düzeyindeki soy hattı yapısını eş zamanlı temel eğriler ile birleştirir [43]; Monocle3 bir temel çizge uyarlar [44]; PAGA gruplar arası komşuluk bağlantılarını özetler [45]. Bağlantılılık bir süreklilikle uyumlu olabilir, ancak farklılaşmayı, yönü veya atasal ilişkiyi bağımsız olarak kanıtlamaz.

R

```r
# Demonstration: choose a biologically justified subset and root first.
library(slingshot)
sce <- as.SingleCellExperiment(so)
root_cluster <- "0"  # replace using independent progenitor-state evidence
stopifnot(root_cluster %in% sce$seurat_clusters)
sce <- slingshot(sce, clusterLabels = "seurat_clusters",
                 reducedDim = "PCA", start.clus = root_cluster)
pseudotime <- slingPseudotime(sce)
```

PYTHON

```python
# Topology exploration on the previously selected neighbour graph.
sc.tl.paga(adata, groups="celltype")
sc.pl.paga(adata, threshold=0.05)
# Optional graph layout for display, not evidence of temporal direction.
sc.tl.draw_graph(adata, init_pos="paga", random_state=42)
```

Temel parametreler

Hücreleri, özellikleri, düşük boyutlu gösterimi, kümelemeyi ve başlangıç kökü seçimini biyolojik soruya göre belirleyin; sonra bu seçimlere duyarlılığı değerlendirin. Yıldız biçimli bir çizge dallanmayı temsil edebilir; bağlantısız bir çizge örnekleme veya analiz seçimlerini yansıtabilir. Hiçbir şekil tek başına biyolojik yörünge varlığını belirlemez. Çıkarım için gerekçelendirilmiş yüksek boyutlu gösterimi tercih edin; UMAP'ın görünür geometrisi uzaklıkları bozabilir. Göreli süreç sıralaması, geçen süre değildir ve köklendirme kanıtı olmadan doğal bir yön taşımaz.

13.2 RNA velocity

RNA velocity, intronları çıkarılmış ve korunmuş RNA miktarlarını bir kinetik modelde kullanarak ifadedeki yerel değişimin yönünü ve hızını tahmin eder [47, 46]. İntronları korunmuş RNA miktarının intronları çıkarılmış RNA miktarına oranı yalnızca uyarlanan ilişki ve model varsayımları çerçevesinde bilgilendiricidir; yüksek oran tek başına bir genin veya soy hattının belirli yönde ilerlediğini kanıtlamaz. USA nicelendirmesi veya uyumlu bir velocyto iş akışı gerekli katmanları sağlayabilir.

PYTHON

```python
# Demonstration after exact barcode/gene harmonisation and UMAP construction.
import scvelo as scv
velocity = sc.read_loom("sample1_velocyto.loom")
assert adata.obs_names.isin(velocity.obs_names).all()
assert adata.var_names.isin(velocity.var_names).all()
matched = velocity[adata.obs_names, adata.var_names]
adata_v = adata.copy()
for layer in ("spliced", "unspliced"):
    adata_v.layers[layer] = matched.layers[layer].copy()
adata_v.X = adata_v.layers["spliced"].copy()
scv.pp.filter_and_normalize(adata_v, min_shared_counts=20, n_top_genes=2000)
scv.pp.moments(adata_v, n_pcs=30, n_neighbors=30)
scv.tl.recover_dynamics(adata_v, n_jobs=1)
scv.tl.velocity(adata_v, mode="dynamical")
scv.tl.velocity_graph(adata_v, n_jobs=1)
scv.pl.velocity_embedding_stream(adata_v, basis="umap")
```

Bu örnek, eşleştirilmiş velocity katmanlarını eklerken mevcut nesnenin UMAP ve üst verisini korur. Katı tanımlayıcı kontrolleri, loom dosyası uyumsuz barkod ön ekleri, gen anahtarları veya farklı özellik kümesi kullanıyorsa bilerek durur; ilerlemeden önce açık bir bire bir eşleme kurun. İşlenmiş nesneyi ilgisiz bir loom nesnesiyle sessizce değiştirip içinde bulunmayan bir yerleştirmeyi çizmeyin. Büyük dinamik model uyarlamaları ölçülmüş bellek/işlemci bütçesi gerektirir; seri işçiler açık bir başlangıç kaynak seçimidir.

Dikkat edilmesi gerekenler

Dinamik model, basit transkripsiyon rejimleri ve intron çıkarılma/RNA yıkım hızları gibi kinetik varsayımlar içerir. Karmaşık düzenleme, yetersiz örnekleme, zayıf kinetik sinyal veya ön işleme bu varsayımları bozabilir. Yalnızca yansıtılmış okları değil, gen düzeyindeki uyumu ve duyarlılığı inceleyin. Yönü model temelli hipotez olarak ele alın; biyolojik soy hattı iddiasından önce bağımsız zaman serisi, müdahale, soy izleme veya belirteç kanıtı arayın. Aynı veriden çıkarılan pseudotime ile uyum, tümüyle bağımsız doğrulama değildir.

14 Tek hücre verileri için şekiller

Bir şeklin yorumu, hangi ifade gösterimini ve hücre etiketlerini yansıttığına bağlıdır. Şekilleri bu nedenle gözden geçirilmiş nesneden üretin ve kullanılan gösterimle etiketleri kaydedin. Bütünleştirme isteğe bağlıdır; şekil, teknik düzeltmenin zorunlu veya biyolojik açıdan zararsız olduğunu ima etmemelidir. Raster PNG ve vektör SVG çıktılarının fiziksel boyutlarını açıkça belirleyin; ardından ikisini de amaçlanan okuma boyutunda inceleyin. Kodun çalışması tek başına okunabilirliği, doğru etiketleri veya bilimsel anlamın korunduğunu göstermez.

R

```r
library(ggplot2)
# Reuse the project's figure theme and font configuration.
save_pair <- function(plot, stem, width, height) {
  ggsave(paste0(stem, ".png"), plot, width = width, height = height,
         units = "in", dpi = 300)
  ggsave(paste0(stem, ".svg"), plot, width = width, height = height,
         units = "in", device = svglite::svglite)
}
```

Bağımsız şekillerde Times New Roman ve opak siyah metin kullanın. Computer Modern/Latin Modern istisnası LaTeX notu ve onun derlenmiş PDF'si için geçerlidir. Renk; hücre tiplerini, koşulları veya ifadeyi kodlamak için kullanılabilir. Kategori eşlemelerini paneller arasında tutarlı tutun; etiket veya biçimlerin

yardımcı olabileceği yerde rengi tek ayırt edici ipucu yapmayın. Açıklamaları genel başlık olarak tekrarlamak yerine şekil dışında verin; gerekli gen adlarını ve panel tanımlayıcılarını koruyun. Kod üç temel şekil çifti üretir; aşağıdaki ek örnekler belirtilen paket ve girdileri gerektirir.

14.1 Hücre tipi ve koşula göre UMAP

İki gösterimin karşılaştırılabilmesi için aynı koordinatları incelenmiş hücre tipine ve koşula göre renklendirin. UMAP, seçilen bir gösterimi özetler; görsel yakınlık veya görünür karışım tek başına koşul etkisini, teknik parti düzeltmesi başarısını veya soy hattı ilişkisi varlığını göstermez.

R

```
p1 <- DimPlot(so, group.by = "celltype", label = TRUE, repel = TRUE) +
  scale_colour_viridis_d() + labs(title = NULL)
p2 <- DimPlot(so, group.by = "condition") +
  scale_colour_viridis_d() + labs(title = NULL)
save_pair(p1 + p2, "umap_celltype_condition", width = 11, height = 5)
```

PYTHON

```
fig = sc.pl.umap(adata, color=["celltype", "condition"], palette="viridis",
                 wspace=0.4, return_fig=True)
fig.set_size_inches(11, 5)
fig.savefig("umap_celltype_condition.png", dpi=300, bbox_inches="tight")
fig.savefig("umap_celltype_condition.svg", bbox_inches="tight")
```

İncelendikten sonra hücre tiplerini açıkça etiketleyin; sayısal küme kimlikleri tekrar çalıştırmalar arasında değişebilir. Nadir popülasyonların görünür kaldığını, etiketleri çakışmadan yerleştirme işleminin noktaları veya metni gizlemediğini ve renk kategorilerinin kayıtlı etiketlerle eşleştiğini denetleyin.

14.2 Belirteç ifade haritaları

İfade haritaları, bir ifade gösterimini yerleştirme üzerine yansıtır. Değerlerin logaritmik normalleştirilmiş ifade mi yoksa başka bir nicelik mi olduğunu belirtin; aynı genin gruplar arasındaki ifadesini karşılaştırırken ortak ölçek kullanın. Birbirinden bağımsız ölçeklenmiş renklerle ilgisiz genleri karşılaştırmaktan kaçının.

R

```
DefaultAssay(so) <- "RNA"  # log-normalized RNA data for expression display
p <- FeaturePlot(so, features = c("CD3D", "MS4A1", "LYZ"), ncol = 3,
                 cols = c("lightgrey", "#440154"), order = TRUE)
p <- p & theme(plot.title = element_text(colour = "black"))  # gene panel labels
save_pair(p, "feature_markers", width = 11, height = 4)
```

PYTHON

```
fig = sc.pl.umap(adata, color=["CD3D", "MS4A1", "LYZ"], ncols=3,
                 cmap="viridis", vmin=0, use_raw=False, return_fig=True)
fig.set_size_inches(11, 4)
fig.savefig("feature_markers.png", dpi=300, bbox_inches="tight")
fig.savefig("feature_markers.svg", bbox_inches="tight")
```

14.3 Kanonik belirteçler için nokta grafikleri ve keman grafikleri

Nokta boyutu saptanma sıklığını, renk ise ortalama ifadeyi kodlayabilir. Çizim işlevinin ortalama alma ve ölçekleme kuralını doğrulayın; ölçeklenmiş renk mutlak ifade ölçümü değildir. Keman grafiği, dağılımı ayrıca gösterebilir. Bu betimleyici özetler hücre tipi anotasyonu kararını destekler; örnek düzeyindeki koşul modelinin yerini almaz.

R

```
markers <- c("CD3D", "CD8A", "MS4A1", "LYZ", "NKG7", "PPBP")
p <- DotPlot(so, assay = "RNA", features = markers, group.by = "celltype") +
  scale_colour_viridis_c() + RotatedAxis() + labs(title = NULL)
save_pair(p, "dotplot_markers", width = 8, height = 5)
```

PYTHON

```
markers = ["CD3D", "CD8A", "MS4A1", "LYZ", "NKG7", "PPBP"]
dot = sc.pl.dotplot(adata, var_names=markers, groupby="celltype",
                    cmap="viridis", return_fig=True)
dot.savefig("dotplot_markers.png", dpi=300, bbox_inches="tight")
dot.savefig("dotplot_markers.svg", bbox_inches="tight")
```

14.4 Örnekler arasında hücre tipi oranları

Koşul içindeki tüm hücreleri havuzlamak yerine her bağımsız örneği çizin. Yığılmış sütun, yakalanmış hücrelerin göreli bileşimini tanımlar; mutlak doku bolluğunu göstermez. Grup karşılaştırmasını değerlendirirken genel görünümün yanında her hücre tipi için örnek düzeyindeki noktaları da göstermeyi değerlendirin.

R

```
prop <- so@meta.data |> dplyr::count(sample, condition, celltype)
p <- ggplot(prop, aes(x = sample, y = n, fill = celltype)) +
  geom_col(position = "fill") + facet_wrap(~condition, scales = "free_x") +
  scale_fill_viridis_d() + labs(x = NULL, y = "Proportion of captured cells")
save_pair(p, "sample_composition", width = 9, height = 5)
```

PYTHON

```
prop = (adata.obs.groupby(["sample", "celltype"], observed=True).size()
        .groupby(level=0).transform(lambda x: x / x.sum()).rename("prop").reset_index())
ax = prop.pivot(index="sample", columns="celltype", values="prop").fillna(0).plot(
    kind="bar", stacked=True, colormap="viridis", figsize=(9, 5))
ax.set_ylabel("Proportion of captured cells")
ax.figure.savefig("sample_composition.png", dpi=300, bbox_inches="tight")
ax.figure.savefig("sample_composition.svg", bbox_inches="tight")
```

14.5 Pseudobulk analizi: volkan grafiği ve ısı haritası

Bir hücre tipi için uyarlanmış örnek düzeyindeki modeli ve açıkça tanımlanmış bir karşılaştırma kullanın. Çoklu test ve etki büyüklüğü seçim kurallarını bildirin. Volkan grafiği içindeki kat değişimi tahminleri ve düzeltilmiş p-değerleri farklı roller üstlenir; sıfır, eksik veya sonlu olmayan değerler sessizce kırpılmak yerine açık çizim kuralıyla ele alınmalıdır.

R

```
# dds is the saved DESeq2 object for this cell type.
res_df <- as.data.frame(results(dds, contrast = c("condition", "treated", "control"))) |>
  tibble::rownames_to_column("feature") |>
  dplyr::mutate(sig = !is.na(padj) & padj < 0.05 & abs(log2FoldChange) > 1)
plot_df <- dplyr::filter(res_df, is.finite(log2FoldChange), is.finite(padj), padj > 0)
p <- ggplot(plot_df, aes(log2FoldChange, -log10(padj), colour = sig)) +
  geom_point(size = 0.8) + scale_colour_manual(values = c("grey70", "#440154")) +
  ggrepel::geom_text_repel(data = dplyr::filter(plot_df, sig), aes(label = feature),
                           max.overlaps = 15, size = 3) +
  labs(x = "log2 fold change (treated / control)", y = "-log10 adjusted p-value")
save_pair(p, "pseudobulk_volcano", width = 7, height = 5)

vsd <- DESeq2::varianceStabilizingTransformation(dds, blind = FALSE)
top_genes <- res_df |> dplyr::filter(sig) |> dplyr::slice_min(padj, n = 30) |>
  dplyr::pull(feature)
if (length(top_genes) > = 2) {
  mat <- assay(vsd)[top_genes, , drop = FALSE]
  mat <- mat[apply(mat, 1, sd) > 0, , drop = FALSE]
  if (nrow(mat) > = 2) {
    ht <- pheatmap::pheatmap(mat, scale = "row", silent = TRUE,
      annotation_col = as.data.frame(colData(dds)[, "condition", drop = FALSE]))
    save_pair(ht$gtable, "pseudobulk_heatmap", width = 7, height = 8)
  }
}
```

Dışlanan sıfır/sonlu olmayan düzeltilmiş p-değerlerini belgeleyin. Isı haritası satır ölçeklemesi gen başına göreli örnek desenlerini gösterir; genler arası bolluğu göstermez. Dönüşümü ve seçim kuralını açıklamada belirtin. Kurala uyan gen sayısı çok azsa istatistiksel olarak anlamlı genlerden ısı haritası üretmek için seçim uydurmayın. Dışa aktarılan DE tabloları anotasyon için kararlı gen kimliklerini korur; nesne anahtarlarını okunabilir etiketlerle değiştirirken doğrulanmış özellik eşlemesini kullanın. Paylaşmadan önce son raster/vektör çıktılarının tamamında kesilme, çakışma ve yazı tipi değişimini inceleyin.

15 Tamamlayıcı bilgiler

Aşağıdaki ayrıntılar birden fazla aşamayı ilgilendirir: Çok örnekli işleme, çekirdekler, büyük veri kümeleri, yeniden üretilebilirlik ve sorun giderme. Bağımsız donör varsayımını sessizce değiştirmeden öğretici iş akışını genişletir.

15.1 Çok örnekli iş akışı

Kütüphane, koşul, donör ve yakalama işlemi ilişkisini örnek tablosunda kurun. Sayım kütüphane başına, kalite kontrolü ve doublet saptama yakalama işlemi başına, koşul çıkarımı ise bağımsız deney birimi başına yapılmalıdır. Bu birimler yalnızca çalışma örneğindeki her donör için bir kütüphane ve bir yakalama işlemi bulunan düzende örtüşür. Teknik yeniden dizileme hatları yeni donör değildir; tekrarlanan donörler, bağımsız sayılan yinelenmiş satırlar yerine uygun bir model gerektirir.

Komut satırı

```
# Explicit illustrative library list; use actual IDs from the reviewed sheet.
for s in control_rep1 control_rep2 control_rep3 treated_rep1 treated_rep2 treated_rep3; do
  bash "$SC_WORKFLOW/scripts/01_count_matrix/01_cellranger_count.sh" "$s"
done
```

Eşlik eden QC aşaması, birleştirmeden önce her örneği işler ve örnek üst verisinin tamamını korur. Aşağıdaki kodlar, önceden işlenmiş örnek nesnelerini birleştirmeyi gösterir; yeni yüklenmiş matrislerde eksik kalite kontrolü ve doublet saptama adımlarını uygulamaz.

R

```r
# Each file is a saved, QC'd singlet object with sample metadata.
samples <- read.delim("config/samples.tsv", colClasses = "character")
obj_list <- lapply(samples$sample, function(s) {
  readRDS(file.path("results/per_sample", paste0(s, ".rds")))
})
so <- merge(obj_list[[1]], obj_list[-1], add.cell.ids = samples$sample)
# Seurat v5 merge retains per-input layers; avoid splitting them again.
```

PYTHON

```python
import anndata as ad
samples = read_samples()
adatas = {s: ad.read_h5ad(f"results/per_sample/{s}.h5ad") for s in samples.index}
# Require identical gene IDs, ordering and processing before concatenation.
reference = next(iter(adatas.values())).var
assert all(a.var.equals(reference) for a in adatas.values())
adata = ad.concat(adatas, label="sample_key", index_unique="-", merge="same")
```

Dikkat edilmesi gerekenler

Bir yöntem yakalama işlemi etiketlerini açıkça alıyorsa birleştirilmiş nesne yakalama işlemi düzeyinde işlenebilir. Hata, birleştirilmiş tüm hücreleri tek yükleme saymak veya her teknik kütüphaneyi bağımsız donör kabul etmektir. Herhangi bir koşul modeli kurmadan önce örnek/hücre ilişkilerini doğrulayın.

15.2 Tek çekirdek RNA dizileme

Çekirdekler, donmuş veya ayrıştırılması zor dokuların analizine olanak sağlayabilir ve çoğu zaman önemli miktarda intronik RNA içerir. Ölçüme uygun referans ve intron sayım politikasını örnekler arasında tutarlı kullanın. Cell Ranger, sürüm 7'den beri intronik okumaları varsayılan olarak içerir; gerçek ayarı ve kütüphane türünü doğrulayın [7].

Çekirdek ve bütün hücre hazırlıkları; RNA dağılımları, saptama desenleri ve bulaşma mekanizmaları açısından farklıdır. Bütün hücre eşiklerini kopyalamak yerine nCount/nFeature dağılımlarını ve ilgili kalite ölçütlerini yeniden değerlendirin. Çekirdeklerde mitokondriyal oranlar sıklıkla düşüktür; beklenmedik yüksek değerler yine de bulaşma veya hazırlama sorunlarını gösterebilir. Beklenen düşük taban düzeyi, ölçütü her durumda yararsız yapmaz. Her veri kümesine aynı düzeltmeyi rutin uygulamak yerine ortam RNA'sı durumunu ve düzeltmenin etkisini değerlendirin.

15.3 Büyük veri kümelerine ölçekleme

Sayım matrisleri seyrek tutulmalı; ölçekleme, model uyarlama ve dönüşüm sırasında oluşabilecek yoğun ara yapılar hesaba katılmalıdır. Büyük verilerde BPCells, disk üzerinde Seurat matrisleri sağlayabilir; HDF5 destekli Bioconductor gösterimleri yüklemeyi erteleyebilir. AnnData'nın disk destekli kipi, bazı bellek içi yüklemeleri sınırlar; ancak her Scanpy işlemini bu yapıyla uyumlu hâle getirmez [3].

R

```r
# Optional: a pre-existing BPCells matrix directory, not a 10x directory.
library(BPCells)
mat <- open_matrix_dir("bpcells_matrix_dir")
so <- CreateSeuratObject(mat)
```

PYTHON

```python
adata_backed = sc.read_h5ad("all_samples.h5ad", backed="r")
# Inspect metadata; materialise only a planned subset when RAM permits.
# adata_small = adata_backed[selected_cells, selected_genes].to_memory()
adata_backed.file.close()
```

Hesaplama arka ucunun ve kullanılan sürümün desteklediği işlemleri seçin. Standart Scanpy QC hesaplamaları veya normalleştirme çağrılarının disk destekli matrisi akış hâlinde işleyeceğini varsaymayın. Belleği ölçün ve gerektiğinde doğrulanmış parçalı işleme akışı kullanın. Yaklaşık komşu yöntemleri hesaplama maliyetini azaltabilir; kullanılabilirlikleri ve ölçeklenmeleri seçilen uygulamaya ve veriye bağlıdır.

15.4 Yeniden üretilebilirlik

Girdileri, referans/anotasyonu, kimyayı, örnek ilişkilerini, hangi hücrelerin tutulduğunu veya dışlandığını ve bu kararların gerekçelerini, parametreleri ve gerçek ortam sürümlerini kaydedin. Doublet simülasyonu, PCA başlangıcı, gerektiğinde komşu arama, kümeleme, UMAP ve model eğitimi gibi rastlantısal işlemlerin yakınında rastgelelik tohumu belirleyin [2]. Tohum bir yeniden üretilebilirlik kaydının parçasıdır; uygulamalar arasında garanti değildir.

R

```r
set.seed(42)
so <- RunUMAP(so, reduction = reduction, dims = dims, seed.use = 42)
set.seed(42)
so <- FindClusters(so, resolution = 0.5, algorithm = 4, random.seed = 42)
sessionInfo()
```

PYTHON

```python
from importlib.metadata import version
sc.tl.umap(adata, random_state=42)
sc.tl.leiden(adata, resolution=0.5, random_state=42,
             flavor="igraph", directed=False, n_iterations=2)
print({p: version(p) for p in ("scanpy", "anndata", "numpy")})
```

Kod, her R aşaması için oturum kayıtları ve Python paket sürümlerini dışa aktarır. Ortam dışa aktarma kodu, adlandırılmış conda ortamını ve etkin analiz yorumlayıcısını gerektirir; dışa aktarım başarısızsa bunu görünür biçimde bildirir. Çözülmüş ortamı ve platformu yerel olarak arşivleyin; taşınabilir kilit dosyası demeden önce geri yüklemeyi doğrulayın. Kodların ayrıştırılması veya küçük sentetik biçim/sayım kontrollerinin geçmesi, biyolojik analizin tam yeniden üretildiğini göstermez.

15.5 Sorun giderme

Belirti	**Olası nedenler**	**Analizi değiştirmeden önce kontrol**
Örnekler UMAP'ta ayrılıyor	Teknik etkiler, gerçek biyoloji, popülasyon farkları veya QC	Tasarımı, belirteçleri ve düzeltilmemiş PC'leri inceleyin; isteğe bağlı bütünleştirmeyi değerlendirin, Bölüm 10
Kümeler mitokondriyal oranı izliyor	Hasarlı hücreler, hazırlama veya gerçek doku/durum değişkenliği	Sembolleri ve örnek başına ölçütleri doğrulayın; filtreleri inceleyin, Bölüm 8
Koşulla ilişkili çok sayıda gen	Güçlü biyoloji, etkilerin birbirine karışması, yalancı tekrar veya sayım girdisi hatası	Bağımsız birimleri, ham sayımları, üst veriyi ve modeli denetleyin, Bölüm 12
Görünür yapı az	Özellik/PC/çizge seçimleri, QC veya gerçek bir süreklilik	Küme dayatmak yerine tanıları inceleyin, Bölüm 9–11
Karışık belirteçli popülasyonlar	Doublet, bulaşma, karışık kümeler veya gerçek durumlar	Puanları, belirteçleri ve örnekler arası tekrarı karşılaştırın, Bölüm 8

Kaynaklar

[1] Luecken MD, Theis FJ. Current best practices in single-cell RNA-seq analysis: a tutorial. *Mol Syst Biol.* 2019;15(6):e8746.

[2] Heumos L, Schaar AC, Lance C, et al. Best practices for single-cell analysis across modalities. *Nat Rev Genet.* 2023;24(8):550–572.

[3] Amezquita RA, Lun ATL, Becht E, et al. Orchestrating single-cell analysis with Bioconductor. *Nat Methods.* 2020;17(2):137–145.

[4] Zheng GXY, Terry JM, Belgrader P, et al. Massively parallel digital transcriptional profiling of single cells. *Nat Commun.* 2017;8:14049.

[5] Macosko EZ, Basu A, Satija R, et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell.* 2015;161(5):1202–1214.

[6] Klein AM, Mazutis L, Akartuna I, et al. Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell.* 2015;161(5):1187–1201.

[7] 10x Genomics. *Cell Ranger.* Software; 2024. `https://www.10xgenomics.com/support/software/cell-ranger`

[8] Kaminow B, Yunusov D, Dobin A. *STARsolo: accurate, fast and versatile mapping/quantification of single-cell and single-nucleus RNA-seq data.* bioRxiv. 2021. doi:10.1101/2021.05.05.442755

[9] He D, Zakeri M, Sarkar H, Soneson C, Srivastava A, Patro R. Alevin-fry unlocks rapid, accurate and memory-frugal quantification of single-cell RNA-seq data. *Nat Methods.* 2022;19(3):316–322.

[10] Lun ATL, Riesenfeld S, Andrews T, et al. EmptyDrops: distinguishing cells from empty droplets in droplet-based single-cell RNA sequencing data. *Genome Biol.* 2019;20:63.

[11] Young MD, Behjati S. SoupX removes ambient RNA contamination from droplet-based single-cell RNA sequencing data. *GigaScience.* 2020;9(12):giaa151.

[12] Yang S, Corbett SE, Koga Y, et al. Decontamination of ambient RNA in single-cell RNA-seq with DecontX. *Genome Biol.* 2020;21:57.

[13] Lun ATL, McCarthy DJ, Marioni JC. A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor. *F1000Research.* 2016;5:2122.

[14] Lun ATL, Bach K, Marioni JC. Pooling across cells to normalize single-cell RNA sequencing data with many zero counts. *Genome Biol.* 2016;17:75.

[15] McCarthy DJ, Campbell KR, Lun ATL, Wills QF. Scater: pre-processing, quality control, normalization and visualization of single-cell RNA-seq data in R. *Bioinformatics.* 2017;33(8):1179–1186.

[16] Germain PL, Lun A, Garcia Meixide C, Macnair W, Robinson MD. Doublet identification in single-cell sequencing data using scDblFinder. *F1000Research.* 2021;10:979.

[17] McGinnis CS, Murrow LM, Gartner ZJ. DoubletFinder: doublet detection in single-cell RNA sequencing data using artificial nearest neighbors. *Cell Syst.* 2019;8(4):329–337.e4.

[18] Satija R, Farrell JA, Gennert D, Schier AF, Regev A. Spatial reconstruction of single-cell gene expression data. *Nat Biotechnol.* 2015;33(5):495–502.

[19] Stuart T, Butler A, Hoffman P, et al. Comprehensive integration of single-cell data. *Cell.* 2019;177(7):1888–1902.e21.

[20] Hao Y, Stuart T, Kowalski MH, et al. Dictionary learning for integrative, multimodal and scalable single-cell analysis. *Nat Biotechnol.* 2024;42(2):293–304.

[21] Wolf FA, Angerer P, Theis FJ. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol.* 2018;19:15.

[22] Virshup I, Rybakov S, Theis FJ, Angerer P, Wolf FA. anndata: Access and store annotated data matrices. *J Open Source Softw.* 2024;9(101):4371.

[23] Hafemeister C, Satija R. Normalization and variance stabilization of single-cell RNA-seq data using regularized negative binomial regression. *Genome Biol.* 2019;20:296.

[24] Korsunsky I, Millard N, Fan J, et al. Fast, sensitive and accurate integration of single-cell data with Harmony. *Nat Methods.* 2019;16(12):1289–1296.

[25] Haghverdi L, Lun ATL, Morgan MD, Marioni JC. Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. *Nat Biotechnol.* 2018;36(5):421–427.

[26] Lopez R, Regier J, Cole MB, Jordan MI, Yosef N. Deep generative modeling for single-cell transcriptomics. *Nat Methods.* 2018;15(12):1053–1058.

[27] Gayoso A, Lopez R, Xing G, et al. A Python library for probabilistic analysis of single-cell omics data. *Nat Biotechnol.* 2022;40(2):163–166.

[28] Luecken MD, Büttner M, Chaichoompu K, et al. Benchmarking atlas-level data integration in single-cell genomics. *Nat Methods.* 2022;19(1):41–50.

[29] McInnes L, Healy J, Melville J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv:1802.03426; 2018.

[30] Van der Maaten L, Hinton G. Visualizing data using t-SNE. *J Mach Learn Res.* 2008;9:2579–2605.

[31] Traag VA, Waltman L, van Eck NJ. From Louvain to Leiden: guaranteeing well-connected communities. *Sci Rep.* 2019;9:5233.

[32] Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. *J Stat Mech Theory Exp.* 2008;2008(10):P10008.

[33] Korsunsky I, Nathan A, Millard N, Raychaudhuri S. *Presto scales Wilcoxon and auROC analyses to millions of observations.* bioRxiv. 2019. doi:10.1101/653253

[34] Aran D, Looney AP, Liu L, et al. Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage. *Nat Immunol.* 2019;20(2):163–172.

[35] Hao Y, Hao S, Andersen-Nissen E, et al. Integrated analysis of multimodal single-cell data. *Cell.* 2021;184(13):3573–3587.

[36] Domínguez Conde C, Xu C, Jarvis LB, et al. Cross-tissue immune cell analysis reveals tissue-specific features in humans. *Science.* 2022;376(6594):eabl5197.

[37] Squair JW, Gautier M, Kathe C, et al. Confronting false discoveries in single-cell differential expression. *Nat Commun.* 2021;12:5692.

[38] Crowell HL, Soneson C, Germain PL, et al. muscat detects subpopulation-specific state transitions from multi-sample multi-condition single-cell transcriptomics data. *Nat Commun.* 2020;11:6077.

[39] Love MI, Huber W, Anders S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol.* 2014;15(12):550.

[40] Robinson MD, McCarthy DJ, Smyth GK. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics.* 2010;26(1):139–140.

[41] Dann E, Henderson NC, Teichmann SA, Morgan MD, Marioni JC. Differential abundance testing on single-cell data using k-nearest neighbor graphs. *Nat Biotechnol.* 2022;40(2):245–253.

[42] Phipson B, Sim CB, Porrello ER, Hewitt AW, Powell J, Oshlack A. propeller: testing for differences in cell type proportions in single cell data. *Bioinformatics.* 2022;38(20):4720–4726.

[43] Street K, Risso D, Fletcher RB, et al. Slingshot: cell lineage and pseudotime inference for single-cell transcriptomics. *BMC Genomics.* 2018;19:477.

[44] Cao J, Spielmann M, Qiu X, et al. The single-cell transcriptional landscape of mammalian organogenesis. *Nature.* 2019;566(7745):496–502.

[45] Wolf FA, Hamey FK, Plass M, et al. PAGA: graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. *Genome Biol.* 2019;20:59.

[46] Bergen V, Lange M, Peidli S, Wolf FA, Theis FJ. Generalizing RNA velocity to transient cell states through dynamical modeling. *Nat Biotechnol.* 2020;38(12):1408–1414.

[47] La Manno G, Soldatov R, Zeisel A, et al. RNA velocity of single cells. *Nature.* 2018;560(7719):494–498.

[48] Wu T, Hu E, Xu S, et al. clusterProfiler 4.0: A universal enrichment tool for interpreting omics data. *The Innovation.* 2021;2(3):100141.

[49] Wolock SL, Lopez R, Klein AM. Scrublet: Computational Identification of Cell Doublets in Single-Cell Transcriptomic Data. *Cell Syst.* 2019;8(4):281–291.e9. https://doi.org/10.1016/j.cels.2018.11.005

[50] Zappia L, Richter S, Ramírez-Suástegui C, et al. Feature selection methods affect the performance of scRNA-seq data integration and querying. *Nat Methods.* 2025;22:834–844. https://doi.org/10.1038/s41592-025-02624-3

[51] Yi C, Cheng J, Chen J, Liu W, Liu J, Li Y. Benchmarking deep learning methods for biologically conserved single-cell integration. *Genome Biol.* 2025;26:398. https://doi.org/10.1186/s13059-025-03869-z

[52] Li S, Lücken M, Marioni JC, Teichmann SA, He P. Toward informed batch correction for single-cell transcriptome integration. *Nat Comput Sci.* 2026;6:123–133. https://doi.org/10.1038/s43588-025-00943-1