A DNA Methylation–based Computational Framework for Tumour–microenvironment State Inference and Molecular Stratification

TUTORIAL HANDBOOK

GEO Series Cancer Cohorts

WITTEN BY

Saltiel Hamese, M.Sc.

DATE

January 2026

This handbook forms part of the work completed by Saltiel Hamese during his Doctoral studies.

The work was supported by the South African Government via the National Skills Development Fund (NSDF), with administrative oversight provided by the National Research Foundation (NRF).

AUTHORS AND AFFILIATIONS

Saltiel Hamese.[1,2], Mutsa Takundwa.[1], Earl Prinsloo.[2], Deepak Balaji Thimiri Govinda Raj[1]

[1]Synthetic Nanobiotechnology and Biomachines, Synthetic Biology and Precision Medicine Centre, Council for Scientific and Industrial Research (CSIR), Pretoria, 0002, South Africa.

[2]Biotechnology Innovation Centre, PO Box 94, Rhodes University (Rhodes), Makhanda, 6140 South Africa

CORRESPONDING AUTHOR: Deepak Balaji Thimiri Govinda Raj[1]

EXECUTIVE SUMMARY

This handbook accompanies the research article *"A DNA Methylation–Based Computational Framework for Tumour–Microenvironment State Inference and Molecular Stratification"* and provides a practical, tutorial-style implementation of the analytical framework it introduces. The framework combines statistical programming and machine learning to reconstruct tumour microenvironment (TME) states from high-dimensional molecular data and to align patient tumours with biologically relevant experimental systems, with primary validation based on genome-scale DNA methylation profiling. DNA methylation is used as the core molecular substrate because of its stability and its ability to capture both tumour-intrinsic regulatory programmes and microenvironment-associated signals, including immune and stromal components. The workflow integrates rigorous data harmonisation and quality control, feature-restricted differential methylation analysis, unsupervised tumour state reconstruction, and quantitative tumour–model concordance with predictive modelling, unifying tumour stratification and experimental model selection into a single inference process. This handbook emphasises transparency, reproducibility, and accessibility through fully annotated R workflows and publicly available data, and while centred on epigenomics, the framework is inherently extensible to additional molecular layers, supporting generalisable tumour state inference and precision modelling across cancer types.

BIOINFOMATICS TOPICS

- **Open-Source**

  Modern cancer bioinformatics relies on large—scale, heterogeneous datasets. Open—source (Gene Expression Omnibus, GEO) ensures that analyses are reproducible, auditable, and extendable, enabling other researchers to validate findings and build upon the methodology. Combining open—source analytical tools with machine learning enables scalable, patient—specific, and biologically interpretable insights.

- **Analytical Programming**

  The analytical programming component focuses on structured, reproducible data processing, exploration, and integration of multi-layered molecular and clinical datasets. This ensures that the downstream machine learning analyses are built on clean, well—normalised, and biologically meaningful inputs.

- **Machine Learning (ML)**

  Machine Learning is used to integrate multiple biological signals (DEA, driver, TME) into a predictive model that prioritises the most faithful tumour models per patient. This moves beyond heuristic or single—feature selection, providing a quantitative, reproducible, and patient—specific ranking of models.

- **Precision Oncology**

  Precision oncology is achieved through quantitative, reproducible mapping of patient tumours to model systems, enhanced by machine—learning and contextual molecular features. By emphasizing patient—specific ranks and relative deviations, the approach captures nuances missed by global metrics, improving the relevance of preclinical models for therapy selection.

TABLE OF CONTENTS

BACKGROUND

Cervical cancer remains a persistent yet largely preventable threat to women's health worldwide, with a disproportionate burden borne by women in low- and middle-income countries. In regions such as sub-Saharan Africa, including South Africa, it continues to rank among the leading causes of cancer-related morbidity and mortality despite the availability of screening programmes, vaccination strategies, and treatment modalities. Structural inequities in healthcare access, late-stage diagnosis, and the high prevalence of biologically aggressive disease contribute to poor outcomes, underscoring the need for improved molecular understanding to support more effective and equitable precision medicine strategies for women affected by this disease. At the biological level, a central challenge in cervical cancer management is pronounced intra-tumour heterogeneity (ITH), arising from the coexistence of multiple tumour subclones shaped by genetic variation and tumour microenvironment (TME) pressures. This heterogeneity fuels tumour adaptation, therapeutic resistance, immune evasion, and disease recurrence, thereby limiting the durability of standard treatments and complicating clinical decision-making. These challenges are particularly acute in cervical cancer, which develops within a complex microenvironment influenced by epigenetic regulation, persistent human papillomavirus (HPV) infection, and, in many settings, HIV co-infection, each imposing distinct immune and stromal programmes that fundamentally shape tumour behaviour.

Advances in analytical programming and computational biology have enabled the development of bioinformatics pipelines capable of processing and analysing large-scale patient-derived molecular data, including transcriptomic, epigenomic, proteomic, and genomic profiles. Such pipelines commonly incorporate machine learning approaches for classification, clustering, feature selection, and predictive modelling, with deep learning increasingly applied to image-based and high-dimensional molecular data. However, despite the widespread availability of multi-omics resources through platforms such as The Cancer Genome Atlas (TCGA) and the Gene Expression Omnibus (GEO), their translation into clinically meaningful stratification and therapeutic insight for cervical cancer remains limited. A major barrier is the lack of integrated frameworks that explicitly reconstruct tumour microenvironment states from patient molecular profiles and systematically link these states to experimentally tractable model systems. Compounding this limitation, commonly used cervical cancer models often lack critical immune and stromal components, reducing their ability to faithfully represent patient tumours and contributing to poor translatability of preclinical findings. There is therefore a pressing need for biologically grounded, reproducible computational approaches that infer tumour microenvironment states directly from patient data and connect these states to representative in vitro cancer models, thereby enabling tumour-aware experimental design and advancing precision oncology in cervical cancer.

1. Open-Source Data Retrival

1. Open-Source Data Retrival

Tutorial 1 demonstrates how to programmatically access and structure open-source DNA methylation datasets from the Gene Expression Omnibus (GEO) using the GEOquery R package. The tutorial focuses on retrieving processed methylation data, inspecting sample-level and assay-level metadata, and constructing harmonised patient and cell-line data objects suitable for downstream analyses, including biomarker discovery, tumour–model matching, and drug response inference. The primary patient dataset analysed in this tutorial is GSE279982, which comprises genome-scale DNA methylation profiles from cervical tumour samples collected from HIV-positive Nigerian women, alongside associated clinical metadata. This dataset provides a unique opportunity to investigate tumour microenvironment–linked epigenetic variation in a clinically under-represented population. In addition to patient data, we incorporate a compulsory cervical cancer cell-line methylation dataset, GSE68379, which contains processed methylation profiles for a curated panel of 20 cervical cancer cell lines. Although large pharmacogenomic resources such as the Cancer Cell Line Encyclopedia (CCLE) and Genomics of Drug Sensitivity in Cancer (GDSC) collectively profile over 1,000 cancer cell lines, this tutorial focuses on a targeted and disease-specific cell-line panel. The GSE68379 cell lines are later used for tumour–model concordance analysis, molecular stratification, and downstream drug repurposing workflows, where integration with patient-derived methylation signatures from GSE279982 is essential.

Users may explore these datasets directly via the GEO website (https://www.ncbi.nlm.nih.gov/geo) by searching for GSE279982 and GSE68379 to examine sample annotations, platforms, and supplementary files available for each study. However, all data access and processing in this handbook are performed programmatically to ensure reproducibility and scalability.

Prerequisites

- A working installation of R (version ≥ 4.2)
- Internet access for downloading GEO datasets
- Basic familiarity with R objects, data frames, and matrix operations
- Introductory understanding of DNA methylation data (CpG probes and beta values)

The following R packages are required for this tutorial;

Implementation:

```
#Install packages if not already available
if (!requireNamespace("BiocManager", quietly = TRUE))
install.packages("BiocManager")
BiocManager::install("GEOquery")
install.packages("tidyverse")
#Load libraries
library(GEOquery)
library(tidyverse)
```

1.1. Accessing GEO Data via GEOquery

We use the GEOquery packages to search, access, and download GEO datasets in R. GEOquery streamlines the retieval of processed methylation matrices (e.g., beta-values), sample-level clinical metadata (phenoData), platform annotations (GPL files), and raw data files (IDATs) using getGEOSuppFiles() when required. This ensures that probe annotations, sample clinical information, and assay data remain synchronized throughout downstream analyses.

Retrieving GEO Series (GSE) Objects

We use the getGEO() function to retrieve GEO studies directly into R. This provides metadata, sample annotation tables, and processed methylation matrices when available. For our tutorial, we retrieve both datasets;

Implementation:

```
# Load patient dataset (GSE279982)
gse_id1 <- "GSE279982"
gse1 <- getGEO(gse_id1, GSEMatrix = TRUE)
#Found 1 file(s)
#GSE279982_series_matrix.txt.gz
#|-------------------------------------------------|
#|=================================================|
#|-------------------------------------------------|
#|=================================================|
#|-------------------------------------------------|
#|=================================================|


# Load cell-line dataset (GSE68379)
gse_id <- "GSE68379"
gse <- getGEO(gse_id, GSEMatrix = TRUE)
#Found 1 file(s)
#GSE68379_series_matrix.txt.gz
#|-------------------------------------------------|
#|=================================================|
```

The object returned by getGEO() contains sample-level metadata (phenoData), molecular assay matrices, and platform (GPL) information. These can be inspected and structured for analysis.

1.2. Inspecting and Preparing Clinical Metadata

This section demonstrates how to inspect and extract clinical and sample metadata from GEO methylation studies prior to any downstream preprocessing or analysis. Because GEO objects can contain multiple layers of information—including assay data, phenotype annotations, feature annotations, and platform metadata—it is essential to examine their structure immediately after loading to understand what data are available and how they are organised. Using the patient cohort (GSE279982), and the cancer models (GSE68379), we first inspect the overall structure of each GEO series object to identify the number of samles, available metadata fields, platform annotations, and associated publications. Standard inspection functions such as head(), str(), dim(), and view() are used to confirrm object integrity and reveal the contests of phenoData, featureData, and experimentData;

Implementation:

```
# Inspect overall structure of the patient GEO object
head(gse1)
#$GSE279982_series_matrix.txt.gz
#ExpressionSet (storageMode: lockedEnvironment)
#assayData: 0 features, 576 samples
#  element names: exprs
#protocolData: none
#phenoData
#  sampleNames: GSM8584577 GSM8584578 ... GSM8585152 (576 total)
#  varLabels: title geo_accession ... tissue:ch1 (54 total)
#  varMetadata: labelDescription
#featureData
#  featureNames:
#  fvarLabels: ID SPOT_ID ... SourceSeq (14 total)
#  fvarMetadata: Column Description labelDescription
#experimentData: use 'experimentData(object)'
#  pubMedIds: 40490892
#Annotation: GPL21145


# Inspect overall structure of the cell-line GEO object
head(gse)
#$GSE68379_series_matrix.txt.gz
#ExpressionSet (storageMode: lockedEnvironment)
#assayData: 0 features, 1028 samples
#  element names: exprs
#protocolData: none
#phenoData
#  sampleNames: GSM1669562 GSM1669563 ... GSM1670589 (1028 total)
#  varLabels: title geo_accession ... primary site:ch1 (40 total)
#  varMetadata: labelDescription
#featureData
#  featureNames:
#  fvarLabels: ID Name ... SPOT_ID (37 total)
#  fvarMetadata: Column Description labelDescription
#experimentData: use 'experimentData(object)'
#  pubMedIds: 27397505
#Annotation: GPL13534
```

We then extact the underlying ExpressionSet objects from each GEO series and. Isolate the phenotype metadata tables using pData(). These metadata tables contain sample-level clinical and experimental annotations (e.g., disease status, tissue type, demographic variables, and experimental descriptors);

Implementation:

```
# Extract ExpressionSet objects
eset_patient <- gse1[[1]]
eset_cellline <- gse[[1]]
# Extract phenotype metadata
pheno_patient <- pData(eset_patient)
pheno_cellline <- pData(eset_cellline)
```

At this stage, no filtering, transformation, or normalisation is applied. The goal is strictly to verify data structure, assess metadata completeness, and prepare clean phenotype tables that can be consistently linked to methylation measurements in subsequent tutorials.

1.2.1. Exploring Study Metadata

We know that our main patient methylation dataset is GSE279982. To explore sample annotations.

Implementation:

```
# Check dimensions (samples × variables)
dim(pheno_patient)
#[1] 576  54
```

Similarly, for the cervical cancer cell lines.

Implementation:

```
# Check dimensions (samples × variables)
 dim(pheno_cellline)
#[1] 1028   40
```

1.2.2. Understanding Data Completeness

GEO datasets are heterogenous by design and rarely contain perfectly uniform information across all samples. Within a single GEO Series, samples may differ in the type and completeness of available data, including the presence of processed beta—values, raw IDAT files, multiple experimental runs, or partially annotated clinical variables. This variability is a common feature of public repositories and reflects differences in experimental design, data submission practices, and clinical data availability. Before performing any downstream analyses—such as biomarker discovery, clustering, or tumour—model matching—it is therefore essential to quantify data completeness and identify missing values within both patient and cell—line cohorts. Explicitly assessing missingness allows informed decisions and variable inclusion, filtering thresholds, and appropriate handling of incomplete metadata, thereby reducing

the risk of biased or misleading results. Here we systematically evaluate missing values at both the variable level (per clinical or experimental field) and the sample level (per patient or cell line), ensuring that only well—annotated and analysis—ready data are carried forward;

Implementation:

```
# Patient cohort: missing values per clinical variable
 colSums(is.na(pheno_patient))> # Cell-line cohort: missing values per experimental variable
 colSums(is.na(pheno_cellline))

# Patient cohort: number of missing entries per sample
 pheno_patient$missing_fields <- rowSums(is.na(pheno_patient))


 summary(pheno_patient$missing_fields)
#   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#0.00000 0.00000 0.00000 0.09201 0.00000 2.00000


 # Cell-line cohort: number of missing entries per sample
 pheno_cellline$missing_fields <- rowSums(is.na(pheno_cellline))


 summary(pheno_cellline$missing_fields)
#   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#0.00000 0.00000 0.00000 0.01167 0.00000 2.00000
```

1.3. Retrieving Methylation Data

Unlike TCGA, GEO does not categories datasets by molecular assay type (e.g., DNA methylation or transcriptome profiling). Each GEO Series simply contains samples measured on a specific platform. For methylation studies, GEO often provides either processed beta—value matrices or raw IDAT files in the supplementary section. In this tutorial, we focus on processed beta—values for the patient cohort (GSE279982), while cell—line data (GSE68379) will be reconstructed from raw IDAT files using minfi. Importantly for the patient cohort, we subset samples before loading the beta matrix to disk, allowing scalable analysis of the large GEO dataset and enabling repeated sampling strategies for robustness and sensitivity analysis.

Implementation:

```
# Define file paths
beta_file <- "~/Downloads/GSE279982_average_beta.txt"       # large methylation matrix
sample_list_file <- "~/Downloads/Methylation_Sample_Sets/SampleSet_1_samples.txt"  # selected
samples

# Read sample subset
sel <- readLines(sample_list_file)
message("Loaded ", length(sel), " sample IDs for subset.")
#Loaded 100 sample IDs for subset.
library(data.table)
# Load beta-value subset (ID_REF + selected samples)
beta_subset <- fread(beta_file, select = c("ID_REF", sel))
#|--------------------------------------------------|
#|==================================================|
```

```
message("Beta subset dimensions: ", paste(dim(beta_subset), collapse = " x "))
#Beta subset dimensions: 856366 x 101
# Preview first few CpGs
head(beta_subset[, 1:5])
#       ID_REF 203219670039_R07C01 203219640064_R04C01 203219640023_R03C01
#       <char>               <num>               <num>               <num>
#1: cg18478105          0.01526497         0.008114771          0.00918673
#2: cg09835024          0.00948493         0.017546734          0.02015302
#3: cg14361672          0.73456459         0.613110597          0.69272497
#4: cg01763666          0.94182922         0.934399971          0.85914969
#5: cg12950382          0.92972109         0.814580932          0.92740651
#6: cg02115394          0.02714056         0.020402766          0.02520589
#   203219730145_R08C01
#                 <num>
#1:         0.009334763
#2:         0.024126520
#3         0.571414167
#4:         0.882770343
#5:         0.944727441
#6:         0.007934469
```

Retrieving the cancer model methylation dataset.

Implementation:

```
library(minfi)
# Read IDAT files
rgSet_cells <- read.metharray.exp(base = "~/GSE68379_Cervical_IDATs/idats")
# Convert to beta values
beta_cells <- getBeta(rgSet_cells)
#Loading required package: IlluminaHumanMethylation450kmanifest
# Inspect structure
dim(beta_cells)
#[1] 485512     15
head(beta_cells[, 1:3])
#           GSM1669626_7878191160_R04C01 GSM1669637_7878191160_R03C01
#cg00050873                    0.4620462                    0.5285118
#cg00212031                    0.5019108                    0.5289747
#cg00213748                    0.5704989                    0.4489572
#cg00214611                    0.5913838                    0.5174731
#cg00455876                    0.4337176                    0.4873188
#cg01707559                    0.5089424                    0.5108907
#           GSM1669639_8221924127_R04C01
#cg00050873                    0.5934475
#cg00212031                    0.5502137
#cg00213748                    0.4477612
#cg00214611                    0.5482143
#cg00455876                    0.3475998
#cg01707559                    0.5284069
```

By using sample subsets for patients and reconstructed beta matrices for cell liines, both datasets are now represented in a comparable beta—value format. This enables repeatable analyses such as clustering, biomarker discovery, FGSEA, and tumour—model matching without memory bottlenecks.

1.4. Visualising GEO Query Results

To improve readability and facilitate structured inspection, we convert each GEO Series' phenoData into a tabular format. This allows systemic exploration of available metadata fields and helps identify variables for relevant downstream analyses. Inspecting column names provides and overview of information capture for each sample, including sample identifiers (GSM accessions), experimental details, perform information, and clinical or biological annotations.

Implementation:

```
cat(colnames(pheno_patient), sep = ", ")
#title, geo_accession, status, submission_date, last_update_date, type, channel_count,
#source_name_ch1, organism_ch1, characteristics_ch1, characteristics_ch1.1, characteristics_ch1.2,
#characteristics_ch1.3, characteristics_ch1.4, characteristics_ch1.5, characteristics_ch1.6,
#characteristics_ch1.7, characteristics_ch1.8, characteristics_ch1.9, characteristics_ch1.10,
#molecule_ch1, extract_protocol_ch1, label_ch1, label_protocol_ch1, taxid_ch1, hyb_protocol,
#scan_protocol, description, data_processing, platform_id, contact_name, contact_email,
#contact_phone, contact_department, contact_institute, contact_address, contact_city,
#contact_state, contact_zip/postal_code, contact_country, supplementary_file,
#supplementary_file.1, data_row_count, age:ch1, bmi:ch1, cancer stage:ch1, colposcopy based
#dysplasia grade:ch1, group:ch1, hpv genotype:ch1, participant id:ch1, sentrix_id:ch1, Sex:ch1,
#site:ch1, tissue:ch1> "
```

For the patient cohort (GSE279982), the phenotype table contains extensivee clinical metadata, including age, BMI, cancer stage, HPV genotype, HIV status grouping, tumour site, and Sentix identifiers, alongside standard GEO submisson and processing fields. For the cancer model dataset (GSE68379), the phenotype table primarily captures experimental annotatons, such as cell—line name, COSMIC identifiers, primary histology, tissue of origin, and platform details, reflecting the non-clinical nature of these samples.

Implementation:

```
cat(colnames(pheno_cellline), sep = ", ")
#title, geo_accession, status, submission_date, last_update_date, type, channel_count,
#source_name_ch1, organism_ch1, characteristics_ch1, characteristics_ch1.1, characteristics_ch1.2,
#characteristics_ch1.3, molecule_ch1, extract_protocol_ch1, label_ch1, label_protocol_ch1,
#taxid_ch1, hyb_protocol, scan_protocol, description, data_processing, data_processing.1,
#data_processing.2, platform_id, contact_name, contact_department, contact_institute,
#contact_address, contact_city, contact_state, contact_zip/postal_code, contact_country,
#supplementary_file, supplementary_file.1, data_row_count, cell line:ch1, cosmic_id:ch1, primary
#histology:ch1, primary site:ch1
```

This initial inspection step is critical for identifying relevant variables for clinical stratification and filtering, understanding differences in metadata structure between patient and cell—line datasets, and guiding the construction of harmonised mapping tables for integrative analyses.

1.5. Extracting Key Clinical Variables

Processed methylation matrices often use different identifiers than the clinical metadata. To ensure alignment between beta—value matrices and sample annotations, we create a sample—mapping table that contains all relevant patient—level variables.

Implementation:

```
# Column names in pheno_patient
sentrix_col <- "geo_accession"       # GSM ID
group_col   <- "group:ch1"           # HIV / Cervical Cancer group
hpv_col     <- "hpv genotype:ch1"    # HPV genotype
age_col     <- "age:ch1"             # Patient age
bmi_col     <- "bmi:ch1"             # BMI
stage_col   <- "cancer stage:ch1"    # Cancer stage
site_col    <- "site:ch1"            # Tumour site
clinical_patient <- pheno_patient %>%
    dplyr::select(
        all_of(c(sentrix_col, group_col, hpv_col, age_col, bmi_col, stage_col, site_col))
    ) %>%
    dplyr::rename(
        Patient      = geo_accession,
        HIV_Group    = `group:ch1`,
        HPV_Genotype = `hpv genotype:ch1`,
        Age          = `age:ch1`,
        BMI          = `bmi:ch1`,
        Cancer_Stage = `cancer stage:ch1`,
        Tumour_Site  = `site:ch1`
    )


 # Preview the clinical table
 head(clinical_patient)
#              Patient      HIV_Group HPV_Genotype Age              BMI
#GSM8584577 GSM8584577 HIV-Neg_CC-Pos           NA  62 20.957171162932
#GSM8584578 GSM8584578 HIV-Neg_CC-Pos           NA  70             <NA>
#GSM8584579 GSM8584579 HIV-Neg_CC-Pos           18  38  23.62444749276
#GSM8584580 GSM8584580 HIV-Pos_CC-Pos           45  36               23
#GSM8584581 GSM8584581 HIV-Neg_CC-Pos           16  40               42
#GSM8584582 GSM8584582 HIV-Neg_CC-Pos           16  65 37.105751391466
#           Cancer_Stage Tumour_Site
#GSM8584577          III        JUTH
#GSM8584578          III        JUTH
#GSM8584579            I        JUTH
#GSM8584580           II        JUTH
#GSM8584581          III        JUTH
#GSM8584582           II        JUTH
```

This mapping is crucial for downstream biomarker discovery, clustering, and linking patient methylation profiles to cell-line data for drug response analysis. We do the same for cervical cancer models;

Implementation:

```
# Select relevant columns first
 cellline_cols <- c(
    "geo_accession",        # GSM ID
    "title",                # Sample title
    "source_name_ch1",      # Source / culture info
```

```
    "organism_ch1",         # Species
    "cell line:ch1",        # Cell line symbol
    "primary histology:ch1",# Histology
    "primary site:ch1",     # Tissue / organ
    "cosmic_id:ch1",        # COSMIC ID
    "supplementary_file"    # Links to raw/supplementary files
 )

 # Create full cell-line metadata table

clinical_cellline <- pheno_cellline %>%
    dplyr::select(all_of(cellline_cols)) %>%
    dplyr::rename(
        Cell_GSM     = geo_accession,
        Sample_Title = title,
        Source       = source_name_ch1,
        Organism     = organism_ch1,
        Cell_Symbol  = `cell line:ch1`,
        Histology    = `primary histology:ch1`,
        Tissue_Site  = `primary site:ch1`,
        COSMIC_ID    = `cosmic_id:ch1`,
        Supplementary_File = supplementary_file
    )

 # Filter only cervix-associated lines
 cervix_celllines <- clinical_cellline %>%
    filter(Histology == "cervix")

 # Preview
 head(cervix_celllines)
#             Cell_GSM        Sample_Title     Source     Organism Cell_Symbol
#GSM1669626 GSM1669626       CellLine_BOKU       BOKU Homo sapiens        BOKU
#GSM1669637 GSM1669637     CellLine_C-33-A     C-33-A Homo sapiens      C-33-A
#GSM1669639 GSM1669639      CellLine_C-4-I      C-4-I Homo sapiens       C-4-I
#GSM1669651 GSM1669651     CellLine_CAL-39     CAL-39 Homo sapiens      CAL-39
#GSM1669667 GSM1669667     CellLine_CA-SKI     CA-SKI Homo sapiens      CA-SKI
#GSM1669742 GSM1669742 CellLine_DOTC2-4510 DOTC2-4510 Homo sapiens  DOTC2-4510
#           Histology       Tissue_Site COSMIC_ID
#GSM1669626    cervix urogenital_system    753536
#GSM1669637    cervix urogenital_system    687505
#GSM1669639    cervix urogenital_system    687506
#GSM1669651    cervix urogenital_system    924107
#GSM1669667    cervix urogenital_system    906824
#GSM1669742    cervix urogenital_system    906843
```

1.6. Aligning Beta Matrices with Clinical Metadata

For all downstream analyses, it is essential that beta-value matices and their corresponding metadata tables are perfectly aligned. Specifically, the columns of each beta matrix must correspond exactly to the rows of the associated clinical or experimental metadata, in both content and order. Failure to enforce this alignment can result in incorrect sample labelling and spurious biological interpretation. We therefore explicity subset and order metadata tables to match the columns order of the beta matrices. Because GEO processed methylation matrices are indexed by Sentrix

barcode rather than GSM accession, direct matching to clinical metadata is not possible without an explicit mapping gap. To address this, we extract GSM—Sentrix relationships from the GEO phenotype table and use them to align sampled beta—value columns with their corresponding clinical annotations.

Align the cohort patient metadata with their methylation—processed reads;

Implementation:

```
mapping_table <- pheno_patient %>%
    dplyr::select(
        GSM        = geo_accession,
        Sentrix_ID = `sentrix_id:ch1`
    )
 head(mapping_table)
#                  GSM          Sentrix_ID
#GSM8584577 GSM8584577 203219750070_R05C01
#GSM8584578 GSM8584578 203219750163_R07C01
#GSM8584579 GSM8584579 203219750001_R04C01
#GSM8584580 GSM8584580 203225140068_R04C01
#GSM8584581 GSM8584581 203220070124_R04C01
#GSM8584582 GSM8584582 203219750124_R01C01


 mapping_subset <- mapping_table %>%
    dplyr::filter(Sentrix_ID %in% colnames(beta_subset)[-1])  # exclude ID_REF


 clinical_patient_aligned <- clinical_patient %>%
    dplyr::inner_join(mapping_subset, by = c("Patient" = "GSM")) %>%
    dplyr::arrange(match(Sentrix_ID, colnames(beta_subset)[-1]))


 stopifnot(
    clinical_patient_aligned$Sentrix_ID == colnames(beta_subset)[-1]
 )


 beta_mat <- as.matrix(beta_subset[, -1])
 rownames(beta_mat) <- beta_subset$ID_REF


 head(clinical_patient_aligned)
#     Patient      HIV_Group HPV_Genotype Age             BMI Cancer_Stage
#1 GSM8584639 HIV-Neg_CC-Pos           NA  60 23.875114784206          III
#2 GSM8585033 HIV-Neg_CC-Pos           18  66              27          III
#3 GSM8584661 HIV-Neg_CC-Pos           59  76 20.690494543389            I
#4 GSM8584947 HIV-Neg_CC-Pos           18  55              25          III
#5 GSM8584672 HIV-Neg_CC-Pos           NA  68              30            I
#6 GSM8585032 HIV-Neg_CC-Pos           18  66              27          III
#  Tumour_Site          Sentrix_ID
#1        JUTH 203219670039_R07C01
#2        LUTH 203219640064_R04C01
#3        JUTH 203219640023_R03C01
#4        LUTH 203219730145_R08C01
#5        JUTH 203220070185_R02C01
#6        LUTH 203219640064_R02C01
```

Align cervical cancer models' metadata data with processed beta value.

Implementation:

```
colnames(beta_cells)[1:5]
#[1] "GSM1669626_7878191160_R04C01" "GSM1669637_7878191160_R03C01"
#[3] "GSM1669639_8221924127_R04C01" "GSM1669651_8221932016_R06C01"
#[5] "GSM1669875_8221916156_R03C02"
cell_gsm_from_beta <- stringr::str_extract(
    colnames(beta_cells),
    "^GSM[0-9]+"
 )


 head(cell_gsm_from_beta)
#[1] "GSM1669626" "GSM1669637" "GSM1669639" "GSM1669651" "GSM1669875"
#[6] "GSM1669876"


 cell_beta_map <- tibble(
    Cell_GSM      = cell_gsm_from_beta,
    Beta_Column   = colnames(beta_cells)
 )


 cell_meta_aligned <- cervix_celllines %>%
    dplyr::inner_join(cell_beta_map, by = "Cell_GSM") %>%
    dplyr::arrange(match(Beta_Column, colnames(beta_cells)))


 stopifnot(
    cell_meta_aligned$Beta_Column == colnames(beta_cells)
 )


 beta_cells_mat <- as.matrix(beta_cells)


 head(cell_meta_aligned)
#    Cell_GSM    Sample_Title Source     Organism Cell_Symbol Histology
#1 GSM1669626   CellLine_BOKU   BOKU Homo sapiens        BOKU    cervix
#2 GSM1669637 CellLine_C-33-A C-33-A Homo sapiens      C-33-A    cervix
#3 GSM1669639  CellLine_C-4-I  C-4-I Homo sapiens       C-4-I    cervix
#4 GSM1669651 CellLine_CAL-39 CAL-39 Homo sapiens      CAL-39    cervix
#5 GSM1669875   CellLine_HELA   HELA Homo sapiens        HELA    cervix
#6 GSM1669876 CellLine_HELASF HELASF Homo sapiens      HELASF    cervix
#        Tissue_Site COSMIC_ID
#1 urogenital_system    753536
#2 urogenital_system    687505
#3 urogenital_system    687506
#4 urogenital_system    924107
#5 urogenital_system   1298134
#6 urogenital_system    687509
#
#                   Beta_Column
#1 GSM1669626_7878191160_R04C01
#2 GSM1669637_7878191160_R03C01
#3 GSM1669639_8221924127_R04C01
#4 GSM1669651_8221932016_R06C01
#5 GSM1669875_8221916156_R03C02
#6 GSM1669876_8221932016_R06C02
```

1.7. Sanity Checks Before Analysis

Before proceeding to downstream analyses (e.g., clustering, differential methylation, enrichment analysis, or tumour—model matching), it is essential to confirm that all assay matrices and metadata tables are internally consistent, corrected aligned, and biologically plausible. These sanity checks guard against silent sample misalignment, metadata corruption, and technical artefacts that can invalidate results without generating explicit errors.

The number of samples in each beta matrix must exactly match the number of rows in its corresponding metadata table. This confirms a one-to-one correspondence between samples and annotations.

Implementation:

```
# Cervical cancer patient cohort
 stopifnot(
    ncol(beta_mat) == nrow(clinical_patient_aligned)
 )


 # Cervical cancer models
 stopifnot(
    ncol(beta_cells_mat) == nrow(cell_meta_aligned)
 )
```

We explicitly verify that sample identifiers appear in the same order in both assay and metadata objects. This ensures that every beta—value column maps to correct biological sample.

Implementation:

```
# Cervical cancer patient samples (Sentrix IDs)
 stopifnot(
    identical(
        clinical_patient_aligned$Sentrix_ID,
        colnames(beta_mat)
    )
 )


 # Cervical cancer model samples (GSM-derived columns)
 stopifnot(
    identical(
        cell_meta_aligned$Beta_Column,
        colnames(beta_cells_mat)
    )
 )
```

We confirm that no duplicated samples or unmapped identifiers are present. Duplicate or missing identifiers would indicate a serious data integrity issue.

Implementation:

```
# Cervical cancer Patient cohort
 stopifnot(
    !anyDuplicated(clinical_patient_aligned$Sentrix_ID),
```

```
    !any(is.na(clinical_patient_aligned$Sentrix_ID))
 )

 # Cervical cancer models
 stopifnot(
   !anyDuplicated(cell_meta_aligned$Beta_Column),
    !any(is.na(cell_meta_aligned$Beta_Column))
 )
```

DNA Methylation beta values must lie within the interval [0,1]. Values outside this range typically indicate parsing errors, platform mismatches, or corrupted files. Expected output should fall within or very close to [0, 1].

Implementation:

```
# Cervical cancer patients beta matrix
 range(beta_mat, na.rm = TRUE)
#[1] 1.626024e-05 9.981049e-01


# Cervical cancer models beta matrix
 range(beta_cells_mat, na.rm = TRUE)
#[1] 0 1
```

We assess missing values at the probe and sample levels to ensure downstream methods are not unduly affected. Low and comparable missingness across samples indicates stable preprocessing.

Implementation:

```
# Missing values per cervical cancer patient sample
 summary(colSums(is.na(beta_mat)))
 #  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  #    0       0       0       0       0       0


 # Missing values per cervical cancer model sample
 summary(colSums(is.na(beta_cells_mat)))
 #  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 # 0.000   1.500   2.000   3.333   5.000   9.000
```

For integrative analyses, it is useful to quantify the overlap in CpG probes between patient and cell—line datasets. This shared probe set forms the basis for direct patient—model comparisons.

Implementation:

```
common_cpgs <- intersect(
    rownames(beta_mat),
    rownames(beta_cells_mat)
 )


 length(common_cpgs)
#[1] 446493
```

At this stage, we have verified that assay and metadata dimensions match, sample identifiers are unique, complete, and correctly ordered, beta—values fall within biologically valid ranges, missingness is minimal and well—characterised, and patient and cell—line datasets are directly comparable. Both cohorts are therefore analysis—ready and can be safely used for clustering, biomarker discovery, enrichment analysis, and tumour—model matching in subsequent tutorials.

2. Data Preprocessing, Normalisation & Quality Control

Tutorial 2 describes a reproucible preprocessing and quality—control framework for Illumina 450K/EPC DNA methylation array data, designed to generate harmonised high—confidence beta—value matrices for downstream analyses. The tutorial focuses on identifying and removing technical artefacts that can obsecure biological signal, including low—quality samples, unreliable probes, and platform—specific biases. Here we demostrate how to perform sample-level quality control, probe-level filtering, array normalisation, construct QC-filtered normalised beta—value matrices, and harmonise patient and model datasets. While raw IDAT-based QC is demostrated using cancer model data, the resulting probe filters are consistently applied to both patient cohort, and appropriate cancer models. This ensures that all datasets are reresented in an identical CpG universe, enabling unbiased downstream analyses such as clustering, biomarker discovery, tumour—model matching, and drug-response modelling. All figures generated by pipeline are exported at user specified dpi (i.e., >600 dpi recommended for publication ready quality).

The following pacages will be required for the tutorial;

```
library(minfi)
library(IlluminaHumanMethylation450kmanifest)
library(IlluminaHumanMethylationEPICanno.ilm10b4.hg19)
library(limma)
library(dplyr)
```

Input Data

The following objects are assumed to be available from Tutorial 1;

- `beta_subset → cervical cancer patient beta-value subset (CpGs × selected samples)`
- `beta_mat → cervical cancer patient beta matrix (CpGs × samples, rownames = CpGs)`
- `clinical_patient_aligned → aligned cervical cancer patients metadata`
- `rgSet_cells → raw cancer models IDATs`
- `beta_cells_mat → reconstructed cervical cancer models beta matrix`
- `cell_meta_aligned → aligned cervical cancer models metadata`

Quality Control (QC) Strategy

Quality control is performed at two levels, the sample-level QC (raw IDATs only, where signal intensities are available), and probe-level filtering (applied consistently across datasets). Because processed beta—value lacks raw intensity information, sample—

level QC is restricted to the cell—line cohort, while probe—level filtering is harmonised across both datasets.

1. **Sample-level QC raw IDATs**

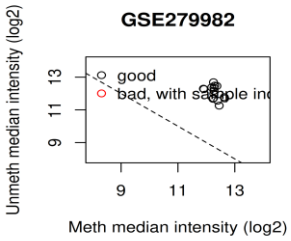
Sample-level quality control requires access to methylated and unmethylated signal intensities. Beacuse, getQC() operates on MethylSet or GenomicMethylSet objects, raw IDAT files must first be converted from RGChanelSet format using preprocessRaw(). This step performs no normalisation and preserves raw signal characteristics for QC assessment.

Implementation:

```
# Convert raw intensities for QC (no normalization)
 mSet_raw <- preprocessRaw(rgSet_cells)


 # Sample-level QC
 qc_cells <- getQC(mSet_raw)
 plotQC(qc_cells)
```

Visualise;

```
tiff(
  filename = "QC_cells_plotQC.tiff",
  width = 1800,     # pixels
  height = 1800,    # pixels
  res = 600,
  compression = "lzw"
)
plotQC(qc_cells)
dev.off()
```



For publication-quality figures, all plot outputs by are exported at 600 dpi using a raster graphics device before plotting. This ensures consistent resolution and compatibility with journal and thesis submission requirements.

2. **Probe-Level Filtering**

To reduce technical bias and improve biological interpretability, we remove probes that are unrealiable or biologically confounding. Probe filtering is derived from cell—line IDATs, then applied to both datasets by extension, to ensure consistency.

Implementation:

```
# Detection P-value Filtering
```

```
 detP <- detectionP(rgSet_cells)

# Keep probes passing detection threshold in all samples
 keep_probes <- rowSums(detP < 0.01) == ncol(detP)

 rgSet_cells_filt <- rgSet_cells[keep_probes, ]
```

We then proceed to remove problematic probes including cross-reactive probes, probes overlapping common SNPs, sex chromosome probes (X/Y), to avoid sex-driven clustering and non-CpG probes;

Implementation:

```
# Extract annotation
anno <- getAnnotation(rgSet_cells_filt)
#Loading required package: IlluminaHumanMethylation450kanno.ilmn12.hg19

#Attaching package: 'IlluminaHumanMethylation450kanno.ilmn12.hg19'

#The following objects are masked from #'package:IlluminaHumanMethylationEPICanno.ilm10b4.hg19':

#    Islands.UCSC, Locations, Manifest, Other, SNPs.132CommonSingle,
#    SNPs.135CommonSingle, SNPs.137CommonSingle, SNPs.138CommonSingle,
#    SNPs.141CommonSingle, SNPs.142CommonSingle, SNPs.144CommonSingle,
#    SNPs.146CommonSingle, SNPs.147CommonSingle, SNPs.Illumina

# Autosomal CpGs only
autosomal <- anno$chr %in% paste0("chr", 1:22)

rgSet_cells_filt <- rgSet_cells_filt[autosomal, ]
```

3. Normalisation

The Noob normalisation method is applied to IDATs, to correct for background fluorescence and dye bias.

Implementation:

```
# Noob normalisation
mSet_cells <- preprocessNoob(rgSet_cells_filt)
# Extract beta values
beta_cells <- getBeta(mSet_cells)
dim(beta_cells)
#[1] 468454     15
```

4. Harmonisation of Datasets

The patient cohort (GSE279982) provides processed beta—values; therefore, IDAT—based normalisation is not required. Instead, we apply the same probe filters derived

from the cell—line data, ensuring both datasets are defined on the same probe universe. To enable direct comparison between tumours and cancer models, both datasets must share an identical set of CG probes;

Implementation:

```
common_probes <- intersect(
    rownames(beta_mat),
    rownames(beta_cells)
 )


 length(common_probes)
#[1] 430885


 beta_patients_h <- beta_mat[common_probes, , drop = FALSE]
 beta_cells_h    <- beta_cells[common_probes, , drop = FALSE]


 stopifnot(
    identical(rownames(beta_patients_h), rownames(beta_cells_h))
 )


 dim(beta_patients_h)
#[1] 430885    100
 dim(beta_cells_h)
#[1] 430885     15
```

Both matrices now contain identical CpG (rows), QC-filtered, normalised beta—values, and samples aligned to validated metadata tables. The step ensures that both datasets are presented in a harmonised CpG space, enabling unbiased clustering, biomarker discovery, and tumour—model matching.

3. Differential Methylation Analysis (limma)

Differential methylation analysis is a key step in understanding how biological groups (e.g., HIV-positive vs. HIV-negative patients) differ at the epigenetic level. Using the high-quality, normalised, QC-filtered, and hamonised methylation matrices generated in Tutorial 2, this tutorial demonstrates how to use limma, one of the most powerful and widely used frameworks for linear modeling in genome-wide methylation studies. Here we prepare M-value matrices (logit-transformed beta-values) for statistical modelling, align phenotype metadata with methylation matrices, construct design matrices for group comparisons, fit probe-wise linear models using limma, extract statistically signifcant differentially methylated possitions (DMPs), visualize DMPs using volcano plots, and finally annotate CpG probes with gene information for biological interpretation.

We perform statistical testing using limma, adjusted p-values for multile testing, extract significant CpGs, and generate marker matrices for downstream intergration with cell-line data.

The following packages are required for this tutorial;

```
#Differential methylation analysis
library(minfi)
library(limma)
library(dplyr)
library(tidyr)
library(ggplot2)
# Methylation utilities and annotation
library(IlluminaHumanMethylationEPICanno.ilm10b4.hg19)
```

Data Preparation

We use the normalized, QC-filtered, harmonized beta-value matrices produced in Tutorial 1. Before running limma, beta-values are converted to M-values (logit transform), which provide better statistical properties for linear modeling.

```
Implementation:

# Convert beta-values to M-values
M_patients <- log2(beta_patients_h / (1 - beta_patients_h))


 # Inspect transformed matrix
 dim(M_patients)
#[1] 430885    100
 head(M_patients[, 1:3])
#           203219670039_R07C01 203219640064_R04C01 203219640023_R03C01
#cg18478105           -6.011439          -6.9334790           -6.752918
#cg09835024           -6.706398          -5.8071144           -5.603488
```

```
#cg14361672           1.468528          0.6642261           1.172752
#cg01763666           4.017099          3.8322719           2.608747
#cg12950382           3.725634          2.1352684           3.675290
#cg02115394          -5.163709         -5.5853520          -5.273265
```

We also ensure that the phenotype metadata aligns with the columns of the M-value matrix;

```
# Confirm alignment
 all(colnames(M_patients) == clinical_patient_aligned$Sample_ID)
#[1] TRUE
```

Design Matrix Creation...stratification

For differential methylation, we define a design matrix that models the effect of HIV status (or other group variables) while optionally adjusting for covariates (eg., age, batch).

Implementation:

```
unique(clinical_patient_aligned$HIV_Group)
#[1] "HIV-Neg_CC-Pos" "HIV-Pos_CC-CIN" "HIV-Pos_CC-Neg" "HIV-Pos_CC-Pos"
 clinical_patient_aligned <- clinical_patient_aligned %>%

    mutate(
        HIV_status = ifelse(grepl("^HIV-Pos", HIV_Group),
                            "Positive", "Negative")
    )


 HIV_group <- factor(
    clinical_patient_aligned$HIV_status,
    levels = c("Negative", "Positive")
 )


 design <- model.matrix(~ HIV_group)
 colnames(design) <- c("Intercept", "HIV_Positive_vs_Negative")


 head(design)
#  Intercept HIV_Positive_vs_Negative
#1         1                        0
#2         1                        0
#3         1                        0
#4         1                        0
#5         1                        0
#6         1                        0
 table(HIV_group)
#HIV_group
#Negative Positive
#      25       75
```

Fitting Linear Models with Limma

We fit linear models for each CpG probe and compute empirical Bayes statistics to identify differentially methylated positions (DMPs);

Implementation:

```
# Fit linear model
 fit <- lmFit(M_patients, design)


 # Empirical Bayes moderation
 fit2 <- eBayes(fit)


 # Inspect top differentially methylated probes
 topTable(fit2, coef = "HIV_Positive_vs_Negative", number = 10)
#                logFC    AveExpr         t      P.Value    adj.P.Val        B
#cg13409704  1.2659441 -2.9914495  6.615296 1.762979e-09 0.0001946588 11.23187
#cg02360862  1.2333498 -1.4116965  6.575168 2.130474e-09 0.0001946588 11.05683
#cg26914296  0.7820629  2.7350705  6.509597 2.900344e-09 0.0001946588 10.77165
#cg00296903 -1.9286253  0.4616419 -6.443763 3.948751e-09 0.0001946588 10.48641
#cg13123165  1.2908212  2.6281263  6.434328 4.126905e-09 0.0001946588 10.44562
#cg20726282 -1.3557811  2.4266508 -6.433732 4.138424e-09 0.0001946588 10.44304
#cg10547025  1.3259367 -0.1032452  6.427596 4.258849e-09 0.0001946588 10.41653
#cg03860256 -1.5818633 -0.7333291 -6.414873 4.519677e-09 0.0001946588 10.36158
#cg12042187  2.1533628  0.5163936  6.402515 4.788094e-09 0.0001946588 10.30825
#cg24363955  1.3197739  3.3742480  6.397741 4.895952e-09 0.0001946588 10.28766
```

Extracting Significant Differential Methylation

We extract significant CpG probes based on ajusted p-values (FDR) and log-fold change thresholds;

```
 Implementation:

# Significance thresholds
 adj_pval_cutoff <- 0.05

 logFC_cutoff <- 0.2


 # Extract significant probes
 sig_probes <- topTable(
    fit2,
    coef = "HIV_Positive_vs_Negative",
    adjust.method = "fdr",
    number = Inf
 ) %>%
    filter(adj.P.Val < adj_pval_cutoff & abs(logFC) > logFC_cutoff)


 # Inspect results
 head(sig_probes)
#                logFC    AveExpr         t      P.Value    adj.P.Val        B
#cg13409704  1.2659441 -2.9914495  6.615296 1.762979e-09 0.0001946588 11.23187
#cg02360862  1.2333498 -1.4116965  6.575168 2.130474e-09 0.0001946588 11.05683
#cg26914296  0.7820629  2.7350705  6.509597 2.900344e-09 0.0001946588 10.77165
#cg00296903 -1.9286253  0.4616419 -6.443763 3.948751e-09 0.0001946588 10.48641
#cg13123165  1.2908212  2.6281263  6.434328 4.126905e-09 0.0001946588 10.44562
#cg20726282 -1.3557811  2.4266508 -6.433732 4.138424e-09 0.0001946588 10.44304
 dim(sig_probes)
#[1] 95378     6
```

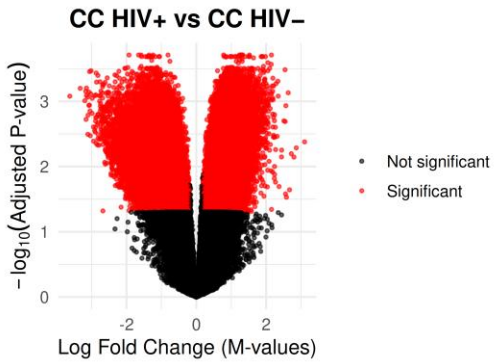
These DMPs can now be used for biomarker discovery, epigenetic clustering, model matching, and drug-response analysis.

Visualization of Differential Methylation

We can visualize DMPs using volcano plots or heatmaps to highlight significant CpGs.

Implementation:

```
# Flag significant probes
sig_probes <- sig_probes %>%
   mutate(Significant = ifelse(adj.P.Val < adj_pval_cutoff, "Yes", "No"))

# Open 600 dpi TIFF device
tiff(
   filename = "Volcano_DMPs_HIV_Positive_vs_Negative.tiff",
   width  = 1800,   # pixels
   height = 1800,   # pixels
   res    = 600,
   compression = "lzw"
)

# Volcano plot
ggplot(sig_probes, aes(x = logFC, y = -log10(adj.P.Val), color = Significant)) +
   geom_point(alpha = 0.6, size = 0.8) +
   theme_minimal(base_size = 12) +
   labs(
       title = "CC HIV+ vs CC HIV-",
       x = "Log Fold Change (M-values)",
       y = expression(-log[10]("Adjusted P-value"))
   ) +
   theme(
       plot.title = element_text(face = "bold", hjust = 0.5),
       legend.title = element_blank()
   )

# Close device
```



Annotating Differentially Methylated Probes

We can link CpG IDs to gene symbols and genomic locations for biological interpretation.

Implementation:

```
# Convert annotation to data.frame
 anno_df <- as.data.frame(anno)


 # Annotate significant probes
 sig_probes_annot <- sig_probes %>%
    mutate(CpG = rownames(sig_probes)) %>%
    left_join(anno_df, by = c("CpG" = "Name"))

 sig_probes_annot_sel <- sig_probes_annot %>%
    select(

        CpG,
        logFC,
        AveExpr,
        t,
        P.Value,
        adj.P.Val,
        chr,
        pos,
        UCSC_RefGene_Name,
        UCSC_RefGene_Group,
        Relation_to_Island,
        Regulatory_Feature_Group
    )


 head(sig_probes_annot_sel)
#         CpG      logFC    AveExpr         t      P.Value    adj.P.Val   chr
#1 cg13409704  1.2659441 -2.9914495  6.615296 1.762979e-09 0.0001946588 chr22
#2 cg02360862  1.2333498 -1.4116965  6.575168 2.130474e-09 0.0001946588  chr1
#3 cg26914296  0.7820629  2.7350705  6.509597 2.900344e-09 0.0001946588 chr10
#4 cg00296903 -1.9286253  0.4616419 -6.443763 3.948751e-09 0.0001946588  chr9
#5 cg13123165  1.2908212  2.6281263  6.434328 4.126905e-09 0.0001946588 chr11
#6 cg20726282 -1.3557811  2.4266508 -6.433732 4.138424e-09 0.0001946588  chr6
#        pos UCSC_RefGene_Name UCSC_RefGene_Group Relation_to_Island
#1  22012495                                                 Island
#2 226411954             MIXL1               Body            S_Shore
#3  74673223              OIT3               Body            OpenSea
#4  90620271                                                 N_Shore
#5 122106236                                                 OpenSea
#6  37693575                                                 OpenSea
#  Regulatory_Feature_Group
#1      Promoter_Associated
#2             Unclassified
#3
#4
#5
#6
```

We now have normalised and QC-filtered M-values for patient tumours, a design matrix modeling HIV-positive versus HIV-negative status, a list of significant diffeentially methylated probes with FDR-adjusted p-values and effect sizes, and finaly annotated DMPs ready for downstream analyses such as biomarker discovery, clustering, and tumour-model correlation.

4. DEA-Restricted ML-Based Tumour-Model Mapping

Global methylation similarity between tumours and models is often dominated by housekeeping CpGs, batch effects, and inter-individual baseline variability. To prioritise biologically meaningful tumour—model concordance, we restrict similarity calculations to differentially methylated probes (DMPs) identified in Tutorial 3. These DMPs represent CpGs associated with patient clinical group HIV status and cervical cancer biology, capturing disease—relevant epigenetic programs. DEA—restricted similarity therefore emphasises pathology-driven variation, improving interpretability and downstream model prioritisation.

The following packages are required for this tutorial:

```
library(dplyr)        # Cleaning, joins, alignment
library(tidyr)        # Reshaping outputs
library(purrr)        # Iteration utilities (optional)
library(ggplot2)      # Visualisation
library(stats)        # Spearman correlation (cor)
```

Data Input Requirements

Tutorial 4 assumes that methylation matrices have already been harmonised and quality controlled. Reusing aligned objects ensures that differences in similarity arise from biological restriction (DMPs) rather than technical inconsistency:

```
beta_cells_h      # Harmonised cell-line beta matrix (CpGs × cell lines)
beta_patients_h  # Harmonised patient beta matrix   (CpGs × patients)
cell_meta_aligned     # Cell-line metadata
clinical_patient_aligned  # Patient metadata
marker_meth  # Optional CpG subset (e.g., DMPs)
```

Align Patient and Model Omics Matrices

Only CpGs present in both datasets can be used for valid correlation analysis. Explicit intersection prevents silent mismatches and ensures comparability:

Implementation:

```
beta_cell    <- as.data.frame(beta_cells_h)
beta_patient <- as.data.frame(beta_patients_h)


beta_cell[]    <- lapply(beta_cell, as.numeric)
beta_patient[] <- lapply(beta_patient, as.numeric)


common_cpgs <- intersect(rownames(beta_cell), rownames(beta_patient))
```

```
 cat("Number of intersecting CpGs:", length(common_cpgs), "\n")
#Number of intersecting CpGs: 430885

beta_cell    <- beta_cell[common_cpgs, , drop = FALSE]
beta_patient <- beta_patient[common_cpgs, , drop = FALSE]
```

Restrict to Differentially Methylated Probes (DEA Output)

DEA restriction removes background CpGs and focuses similarity on HIV-associated methylation changes (In Tutorial 3, we choose HIV clinical group to complete DEA), strengthening clinical signal and reducing noise;

```
 Implementation:

dmp_ids <- rownames(sig_probes)

 common_dmp_cpgs <- intersect(
    dmp_ids,
    intersect(rownames(beta_cell), rownames(beta_patient))
 )

 cat("DEA-restricted CpGs:", length(common_dmp_cpgs), "\n")
#DEA-restricted CpGs: 95378

 beta_cell_dmp    <- beta_cell[common_dmp_cpgs, , drop = FALSE]
 beta_patient_dmp <- beta_patient[common_dmp_cpgs, , drop = FALSE]
```

Compute DEA-Restricted Spearman Correlations

Spearman correlation is robust to non-normality and monotonic relationships, making it well-suited for methylation beta values. Pairwise tumour—model correlations generate a complete similarity landscape;

```
 Implementation:

cor_results_dmp <- data.frame()

 for (cell in colnames(beta_cell_dmp)) {
    for (pat in colnames(beta_patient_dmp)) {

        cor_val <- suppressWarnings(
            cor(
                beta_cell_dmp[, cell],
                beta_patient_dmp[, pat],
                method = "spearman",
                use = "pairwise.complete.obs"
            )
        )

        cor_results_dmp <- rbind(
            cor_results_dmp,
            data.frame(
                CellLine    = cell,
```

```
                Patient     = pat,
                Correlation = cor_val
            )
        )
    }
}
```

Annotate Correlations with Clinical and Model Metadata

Annotation enables stratified visualisation and biological interpretation, linking similarity patterns to HIV status, HPV genotype, and clinical stage, and mapping GSM IDs to interpretable model names;

Implementation:

```
# Annotate with patient metadata
cor_results_dmp_annotated <- cor_results_dmp %>%
    left_join(
        clinical_patient_aligned %>%
            select(Sentrix_ID, HIV_status, HPV_Genotype, Age, BMI, Cancer_Stage),
        by = c("Patient" = "Sentrix_ID")
    ) %>%
    left_join(
        cell_meta_aligned %>% select(Beta_Column, Cell_Symbol),
        by = c("CellLine" = "Beta_Column")
    ) %>%
    mutate(
        CellLine = ifelse(is.na(Cell_Symbol), CellLine, Cell_Symbol)
    ) %>%
    select(-Cell_Symbol)


head(cor_results_dmp_annotated)
#  CellLine             Patient Correlation HIV_status HPV_Genotype Age
#1     BOKU 203219670039_R07C01   0.4796670   Negative           NA  60
#2     BOKU 203219640064_R04C01   0.5046660   Negative           18  66
#3     BOKU 203219640023_R03C01   0.4776081   Negative           59  76
#4     BOKU 203219730145_R08C01   0.5634049   Negative           18  55
#5     BOKU 203220070185_R02C01   0.5808074   Negative           NA  68
#6     BOKU 203219640064_R02C01   0.6273229   Negative           18  66
#              BMI Cancer_Stage
#1 23.875114784206          III
#2              27          III
#3 20.690494543389            I
#4              25          III
#5              30            I
#6              27          III
```

Visualize DEA-restricted Similarity

DEA—restricted Spearman correlations derived from the top differentially methylated probes identified in Tutorial 3 were visualised to examine tumour—model similarity structure. Restricting similarity calculations to disease—associated CpGs reduces background noise from non—informative methylation sites and highlights biologically relevant epigenetic variation. Heatmaps, Multi—line similarity profiles, and metadata—stratified visualisations provide an interpretable link between patent—level epigenetic

heterogeneity and tumour—model mapping, supporting downstream model prioritization for functional and translational analyses.

1. Heatmap top DEA-Restricted DMPs

To assess whether HV-associated methylation captures coherent epigenetic structure, we visualise the top 500 DMPs across patients. CpG β-values were scaled per probe, and patients were annotated by clinical and demographic variables:

Implementation:

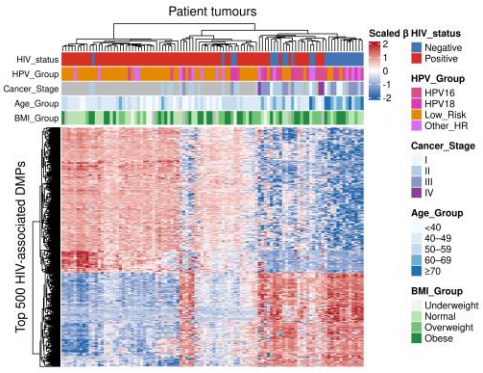
```
top_dmp_ids <- rownames(sig_probes %>% arrange(adj.P.Val))[1:500]


beta_top_dmp <- beta_patient_dmp[top_dmp_ids, , drop = FALSE]
beta_top_dmp_scaled <- t(scale(t(beta_top_dmp)))
```

```
> top_dmp_ids <- rownames(sig_probes %>% arrange(adj.P.Val))[1:500]
>
> beta_top_dmp <- beta_patient_dmp[top_dmp_ids, , drop = FALSE]
> beta_top_dmp_scaled <- t(scale(t(beta_top_dmp)))
```

```
#Ensure column names match patient IDs

stopifnot(all(colnames(beta_top_dmp_scaled) %in% clinical_patient_aligned$Sentrix_ID))

patient_anno <- clinical_patient_aligned %>% filter(Sentrix_ID %in%
colnames(beta_top_dmp_scaled)) %>% distinct(Sentrix_ID, .keep_all = TRUE) %>%
column_to_rownames("Sentrix_ID") %>% .[colnames(beta_top_dmp_scaled), ] # enforce identical order

anno_colours <- list(


    HIV_status = c(
        "Positive" = "#D73027",
        "Negative" = "#4575B4"
    ),


    HPV_Group = c(
        "HPV16"    = "#1B9E77",
        "HPV18"    = "#D95F02",
        "Other_HR" = "#7570B3",
        "Low_Risk" = "#E7298A",
        "Unknown"  = "grey80"
    ),


    Cancer_Stage = c(
        "I"       = "#EDF8FB",

        "II"      = "#B3CDE3",
        "III"     = "#8C96C6",
        "IV"      = "#88419D",
        "NA" = "grey85"

    ),
    Age_Group = c(
        "<40"   = "#F7FBFF",
        "40-49" = "#DEEBF7",
        "50-59" = "#C6DBEF",
```

```
        "60-69" = "#6BAED6",
        "≥70"   = "#2171B5"
    ),

    BMI_Group = c(
        "Underweight" = "#EDF8E9",
        "Normal"      = "#BAE4B3",
        "Overweight"  = "#74C476",
        "Obese"       = "#238B45"
    )
)


patient_anno <- patient_anno %>%
    mutate(
        Age = as.numeric(Age),
        BMI = as.numeric(BMI),

        Age_Group = cut(
            Age,
            breaks = c(0, 40, 50, 60, 70, Inf),
            labels = c("<40", "40-49", "50-59", "60-69", "≥70"),
            right = FALSE
        ),

        BMI_Group = cut(
            BMI,
            breaks = c(0, 18.5, 25, 30, Inf),
            labels = c("Underweight", "Normal", "Overweight", "Obese"),
            right = FALSE
        ),

        HPV_Genotype = as.character(HPV_Genotype),

        HPV_Group = case_when(
            HPV_Genotype == "16" ~ "HPV16",
            HPV_Genotype == "18" ~ "HPV18",
            HPV_Genotype %in% c(
                "31","33","35","39","45","51","52","56","58","59","68","73","82"
            ) ~ "Other_HR",
            is.na(HPV_Genotype) ~ "Unknown",
            TRUE ~ "Low_Risk"
        ),

        Cancer_Stage = ifelse(is.na(Cancer_Stage), "NA", Cancer_Stage)
    )

ha <- HeatmapAnnotation(
    df = patient_anno %>%
        select(
            HIV_status,
            HPV_Group,
            Cancer_Stage,
            Age_Group,
            BMI_Group
```

```
        ),
    col = anno_colours,
    annotation_name_side = "left",
    annotation_name_gp = grid::gpar(fontsize = 9)
)

tiff(
    "Heatmap_Top500_DMPs_Clinical_Annotations.tiff",
    width = 4200,
    height = 3200,
    res = 600,
    compression = "lzw"
)

Heatmap(
    beta_top_dmp_scaled,
    name = "Scaled β",
    top_annotation = ha,
    show_row_names = FALSE,
    show_column_names = FALSE,
    cluster_rows = TRUE,
    cluster_columns = TRUE,
    row_title = "Top 500 HIV-associated DMPs",
    column_title = "Patient tumours",
    col = colorRamp2(
        c(-2, 0, 2),
        c("#2166AC", "white", "#B2182B")
    )
)

dev.off()
```

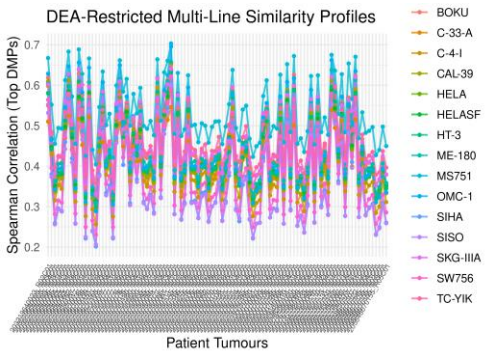


2. Multi-line DEA-restricted similarity profiles

Multi—line plots visualise how each model behaves across the full patient cohort, identifying consistently high—performing models, patient—specific concodance, and variability masked by summary statistics.

Implementation:

```
cor_results_dmp_annotated <- cor_results_dmp_annotated %>%
    mutate(

        Patient = as.character(Patient),
        CellLine = as.character(CellLine),
        Age = as.numeric(Age),
        BMI = as.numeric(BMI),
        Correlation = as.numeric(Correlation)
    )


 library(ggplot2)


 tiff(
    "DEA_Restricted_MultiLine_Similarity.tiff",
    width = 3500,
    height = 2500,
    res = 600,
    compression = "lzw"
 )


 ggplot(cor_results_dmp_annotated, aes(x = Patient, y = Correlation, group = CellLine, color =
CellLine)) +
    geom_line(alpha = 0.8, linewidth = 0.7) +
    geom_point(size = 1) +
    labs(
        title = "DEA-Restricted Multi-Line Similarity Profiles",
        x = "Patient Tumours",
        y = "Spearman Correlation (Top DMPs)"
    ) +
    theme_minimal(base_size = 12) +
    theme(
        axis.text.x = element_text(angle = 60, hjust = 1, size = 6),
        legend.position = "right"
    )


 dev.off()
```



Thus far, this tutorial demostrated how restricting tumour—model similarity to DEA—derived differentially methylated probes improves biological interppretabiliity by focusing on diease-associated epigenetic variation. DEA—restricted Spearman correlations revealed structured tumour—model relationships that were obsecured in genome—wide

similarity analyses, and visualisation through heatmaps and multi—line profiles highlighted both cohort—level patterns and patient—specific heterogeneity.

DEA-Restricted ML-Based Tumour—Model Mapping

DEA-restricted methylation similarity provides a disease—informed representation of tumour—model concordance by focusing only on biologically relevant DMPs. However, absolute Spearman correlations alone are insufficient for prioritising optimal models. Two patients can have similar absolute correlations but very different model specificity, what matters biologically is how exceptional a model is relative to all others for the same patient, this motivates within-patient normalisation and relative ranking—based ML. We therefore integrate absolute and relative similarity using a machine—learning ranking framework, moving beyond naive correlation pairs.

Requirements

`cor_results_dmp_annotated must contain:`

- `Patient`
- `CellLine`
- `Correlation (DEA-restricted Spearman)`

1. Input Data Preparation

Each observation must represent a single patient—model pairing, forming the atomic unit for learning. At this stage, we restrict features to DEA-restricted Spearman correlations only, ensuring the ML task remains interpretable and biologically grounded.

`Implementation:`

```
# cor_results_dmp_annotated must contain: CellLine, Patient, Correlation (DEA-restricted)
 ml_features <- cor_results_dmp_annotated %>%
    select(Patient, CellLine, DEA_Cor = Correlation)
```

2. Within—Patient Normalisation (Relative Similarity)

Absolute correlation values are not directly comparable across patients. We therefore normalise within each patient, capturing how well a model performs relative to all other models tested for that tumour. Two complementary representations are used DEA_Rank and DEA_Z, respectively representing ordinal model ranking per patient, and standardised deviation from patient—specific mean similarity. This ensures that learning is driven by relative concordance, not cohort—wide magnitude effects.

`Implementation:`

```
ml_features <- ml_features %>%
    group_by(Patient) %>%
    mutate(
        DEA_Rank = rank(-DEA_Cor, ties.method = "average"),
        DEA_Z = scale(DEA_Cor)
    ) %>%
    ungroup()
```

3. Defining Training Labels

Because no ground-truth “best model” exists, we use weak supervision, where top-performing (n=3) DEA-restricted models per patient, and remaining models, respectively represent the positive class (Lable = 1), and negative class (Label = 0).

Implementation:

```
ml_features <- cor_results_dmp_annotated %>%
    select(Patient, CellLine, DEA_Cor = Correlation) %>%
    group_by(Patient) %>%
    mutate(
        DEA_Rank = rank(-DEA_Cor, ties.method = "average"),
        DEA_Z    = as.numeric(scale(DEA_Cor))
    ) %>%
    ungroup() %>%
    mutate(
        Label = factor(
            ifelse(DEA_Rank <= 3, "Yes", "No"),
            levels = c("No", "Yes")   # "Yes" is positive class
        )
    )
 str(ml_features$Label)
# Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 2 2 1 ...
 table(ml_features$Label)

#  No  Yes
#1200  300
```

4. ML-Model Selection

Initial logistic regression models failed to converge due to near-perfect separation induced by ranking features, strong colinearity between DEA_Cor and DEA_Rank, binary labels derived from ranks themselves. Random Forest are better suited because they handle non-linear decision boundaries, are robust to correlated predictors, do not rely on distributional assumptions, and naturally capture threshold—like ranking behaviour;

Implementation:

```
library(caret)
set.seed(123)


train_idx <- createDataPartition(
    ml_features$Label,
    p = 0.7,
    list = FALSE
)


train_data <- ml_features[train_idx, ]
test_data  <- ml_features[-train_idx, ]
```

```
rf_model <- train(
  Label ~ DEA_Cor + DEA_Z + DEA_Rank,
  data = train_data,
  method = "rf",
  trControl = trainControl(
    method = "cv",
    number = 5,
    classProbs = TRUE,
    summaryFunction = twoClassSummary
  ),
  metric = "ROC",
  tuneLength = 5
)
```

5. ML-Model Evaluation

Performance is evaluated using ranking—aware metrics, rather than raw accuracy. The ability to disntiguish top vs non-top models, the correctness of top—ranked models, and the consistency of selected models, are respectively evaluated using ROC-AUC, Precision@K, and Stability across patients;

Implementation:

```
 pred <- predict(rf_model, test_data, type = "prob")
 roc_obj <- pROC::roc(test_data$Label, pred$Yes)
#Setting levels: control = No, case = Yes
#Setting direction: controls < cases
 auc(roc_obj)
#Area under the curve: 1
```

```
rf_model
pred_probs <- predict(
  rf_model,
  test_data,
  type = "prob"
)[, "Yes"]
```

6. Model-Derived Tumour—Model Ranking

The trained model outputs a probability of being a top model, which serves as a learned similarity score integrating absolute DEA-restricted correlation, relative deviation within patient, and rank—based dominance. This produces a patient—specific ranked list of models, replacing heuristic selection.

Implementation:

```
ml_features$ML_Score <- predict(
  rf_model,
  ml_features,
  type = "prob"
)[, "Yes"]


ml_ranked <- ml_features %>%
  group_by(Patient) %>%
```

```
  arrange(desc(ML_Score)) %>%
  mutate(ML_Rank = row_number()) %>%
  ungroup()
```

7. Visualisations

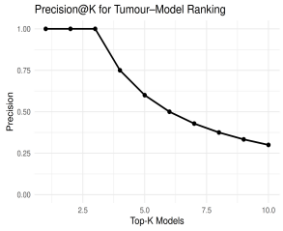
7.1. Precision@K for Tumour—Model Ranking

The Precision@K plot evaluates how well the model identifies the top—ranked models per patient. Instead of overall accuracy, which is insensitive to rank, this metric focuses on whether the models the algorithm prioritises (top k) truly correspond to the “yes” labels (top n models by weak supervision). This is particularly relevant in biomedical contexts, where selecting a few high—confidence models per patient is more important than general classification performance.

Implementation:

```
library(dplyr)

 K_vals <- 1:10

 precision_at_k <- function(df, k) {
    df %>%
        group_by(Patient) %>%
        arrange(desc(ML_Score)) %>%
        slice_head(n = k) %>%
        summarise(Prec = mean(Label == "Yes")) %>%
        pull(Prec) %>%
        mean(na.rm = TRUE)
 }

 prec_df <- data.frame(
    K = K_vals,
    Precision = sapply(K_vals, function(k) precision_at_k(ml_ranked, k))
 )

 tiff(
    "Precision_at_K_Model_Selection.tiff",
    width = 3200, height = 2400, res = 600, compression = "lzw"
 )

 ggplot(prec_df, aes(K, Precision)) +
    geom_line(linewidth = 1) +
    geom_point(size = 2) +
    scale_y_continuous(limits = c(0, 1)) +
    labs(
        title = "Precision@K for Tumour-Model Ranking",
        x = "Top-K Models",
        y = "Precision"
    ) +
    theme_minimal(base_size = 12)
```

```
dev.off()
```

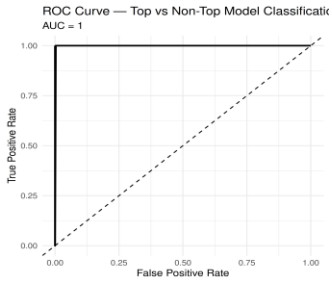


7.2. ROC Curve—Top vs Non—Top Model Classification

The ROC curve evaluates the model's ability to distinguish Top vs non—top models across all patients. By plotting the true positive rate (sensitivity) against the false positive rate (1—specificity), it visualises classification performance across all thresholds. The AUC (Area Under the Curve) summarises performance in a single number: higher AUC indicates better ranking—aware classification.

Implementation:

```
library(pROC)
library(ggplot2)

pred_probs <- predict(rf_model, test_data, type = "prob")[, "Yes"]
roc_obj <- roc(test_data$Label, pred_probs, levels = c("No", "Yes"))
Setting direction: controls < cases

roc_df <- data.frame(
   TPR = roc_obj$sensitivities,
   FPR = 1 - roc_obj$specificities
)

tiff(
   "ROC_Curve_TME_Model_Selection.tiff",
   width = 3000, height = 3000, res = 600, compression = "lzw"
)

ggplot(roc_df, aes(FPR, TPR)) +
   geom_line(linewidth = 1) +
   geom_abline(linetype = "dashed") +
   labs(
       title = "ROC Curve – Top vs Non-Top Model Classification",
       subtitle = paste("AUC =", round(auc(roc_obj), 3)),
       x = "False Positive Rate",
       y = "True Positive Rate"
   ) +
   theme_minimal(base_size = 12)

dev.off()
```

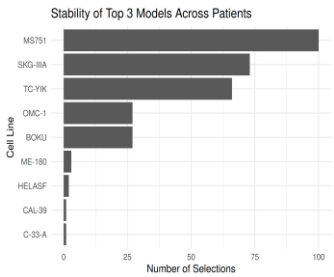


7.3. Stability of Top—K Models Across Patients

This plot assesses consistency of model prioritisation across the patient cohort. By counting how often each cell line appears among the top—k predictions for all patients, it quantifies stability and identifies universally strong models. This complements the Precision@K and ROC metrics by revealing whether certain models are repeatedly recommended.

Implementation;

```
top_k <- 3

stability_df <- ml_ranked %>%
   filter(ML_Rank <= top_k) %>%
   count(CellLine, name = "Selection_Count") %>%
   arrange(desc(Selection_Count))

tiff(
   "TopModel_Stability_Across_Patients.tiff",
   width = 3600, height = 2600, res = 600, compression = "lzw"
)

ggplot(stability_df, aes(reorder(CellLine, Selection_Count), Selection_Count)) +
   geom_col() +
   coord_flip() +
   labs(
       title = paste("Stability of Top", top_k, "Models Across Patients"),
       x = "Cell Line",
       y = "Number of Selections"
   ) +
   theme_minimal(base_size = 12)

dev.off()
```

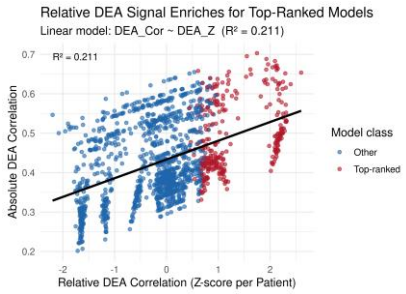


Absolute vs Relative Correlation Regression

To formally demostrate that relative concorrdance carries additional signal, we regress absolute DEA correlation, and against patient—specific relative z-scores, this justifies the ML framework empirically.

Implementation:

```
#ml_features already contains:Patient, CellLine, DEA_Cor (absolute), DEA_Rank, DEA_Z (scaled per
patient)
reg_data <- ml_features %>% select(Patient, CellLine, DEA_Cor, DEA_Z)

reg_model <- lm(DEA_Cor ~ DEA_Z, data = reg_data)
summary(reg_model)
reg_lm <- lm(DEA_Cor ~ DEA_Z, data = reg_data)
r2_val <- summary(reg_lm)$r.squared

tiff("ML_Relative_vs_Absolute_Correlation.tiff",
     width = 3500, height = 2500, res = 600, compression = "lzw")

reg_data <- ml_features %>%
  mutate(Top_Model = ifelse(DEA_Rank <= 3, "Top-ranked", "Other"))


 ggplot(reg_data, aes(x = DEA_Z, y = DEA_Cor, colour = Top_Model)) +
    geom_point(alpha = 0.6, size = 1.4) +
    geom_smooth(method = "lm", se = FALSE, colour = "black") +
    scale_colour_manual(
        values = c(
            "Top-ranked" = "#B2182B",
            "Other"      = "#2166AC"
        )
    ) +
    theme_minimal(base_size = 12) +
    labs(
        title = "Relative DEA Signal Enriches for Top-Ranked Models",
        subtitle = paste0("Linear model: DEA_Cor ~ DEA_Z  (R² = ",
                          round(r2_val, 3), ")"),
        x = "Relative DEA Correlation (Z-score per Patient)",
        y = "Absolute DEA Correlation",
        colour = "Model class"
    ) +
    annotate(
        "text",
        x = min(reg_data$DEA_Z, na.rm = TRUE),
        y = max(reg_data$DEA_Cor, na.rm = TRUE),
        hjust = 0,
        vjust = 1,
        size = 3.5,
        label = paste0("R² = ", round(r2_val, 3))
    )
#`geom_smooth()` using formula = 'y ~ x'
```



This tutorial concludes by demonstrating that absolute correlation values alone are insufficient for tumour–model prioritisation, and that within-patient normalisation captures how exceptional a model is relative to the full model landscape. By integrating absolute DEA-restricted correlations with relative ranking and deviation metrics in a Random Forest framework, we derived patient-specific, data-driven tumour–model rankings that are robust to collinearity and rank-induced separation. Regression analysis further confirmed that relative similarity contributes additional signal beyond absolute correlation. An R^2 of 0.211 indicates that within-patient DEA-scaled correlations explain approximately 21% of the variance in absolute correlations across the cohort. This reflects meaningful added information from relative scaling, while also highlighting substantial residual variability driven by inter-patient baseline differences, technical effects, and biological heterogeneity. Together, these results motivate the integration of additional similarity dimensions, such as tumour microenvironment (TME) and driver-associated signals, in downstream ML-based matching frameworks.

5. Driver-Restricted Tumour—BioModel Mapping

The following R packages will be used in this tutorial;

```
library(dplyr)
library(tidyr)
library(IlluminaHumanMethylationEPICanno.ilm10b4.hg19)
library(readr)
```

Prerequisites

```
beta_cells_h        # Harmonised cell-line β-matrix
beta_patients_h     # Harmonised patient β-matrix
driver_cpgs         # EPIC-annotated CpGs mapping to driver genes
design              # DEA design matrix (from Tutorial 3)
fit                 # limma model fit (from Tutorial 3)
```

5.1) Driver-Restricted Tumour—BioModel Mapping

5.1.1) Import Cancer Driver Gene List

Cancer driver genes were obtained from the Cell Model Passports (CMP) resource using the driver_genes_20241212 dataset, which contains curated, biologically validated driver alterations across cancer models. We then poduce a final curated cancer driver gene panel:

Implementation:

```
#Load cancer driver gene list
 driver_genes <- read_csv("driver_genes_20241212.csv")
#Rows: 791 Columns: 7
#── Column specification ────────────────────────────────────────────────────────
#Delimiter: ","
#chr (7): symbol, method_of_action, cosmic_moa, intogen_moa, huang_et_al_moa,...

#i Use `spec()` to retrieve the full column specification for this data.
#i Specify the column types or set `show_col_types = FALSE` to quiet this message.

 driver_symbols <- driver_genes$symbol %>%
    unique() %>%
    sort()
 length(driver_symbols)
#[1] 791
```

5.1.2) Map Driver Genes to EPIC-array CpGs

Driver genes are mapped to CpGs using the EPIC array annotation, generating a driver—specific CpG panel.

Implementation:

```
# Load EPIC array annotation
 anno <- getAnnotation(IlluminaHumanMethylationEPICanno.ilm10b4.hg19)

 # Convert Bioconductor DFrame → data.frame
 anno_df <- as.data.frame(anno)

 # Subset annotation to driver genes
 driver_cpgs <- anno_df %>%
    filter(UCSC_RefGene_Name %in% driver_symbols) %>%
    select(Name, UCSC_RefGene_Name, chr, pos)

 # Inspect
 head(driver_cpgs)
#                 Name UCSC_RefGene_Name   chr       pos
#cg14428027 cg14428027             BMPR2  chr2 203241027
#cg02980693 cg02980693             GATA2  chr3 128208970
#cg02550231 cg02550231              CDX2 chr13  28542287
#cg24666276 cg24666276               MYC  chr8 128748214
#cg04575263 cg04575263             RBM15  chr1 110882079
#cg27535237 cg27535237             STK11 chr19   1220632
> nrow(driver_cpgs)
#[1] 13136
```

5.1.3) Construct Driver-Restricted Methylation Matrices

Using harmonised, QC-filtered β-matrices from earlier tutorials, we subset to CpGs present in both patients and models. The resulting matrices capture driver—focused methylation profiles suitable for DEA-restricted similarity analysis.

Implementation:

```
driver_cpg_ids <- driver_cpgs$Name
 common_driver_cpgs <- intersect(
    intersect(driver_cpg_ids, rownames(beta_cell)),
    rownames(beta_patient)
 )

 beta_cells_driver    <- beta_cell[common_driver_cpgs, , drop = FALSE]
 beta_patients_driver <- beta_patient[common_driver_cpgs, , drop = FALSE]

 dim(beta_cells_driver)
#[1] 8023   15
 dim(beta_patients_driver)
#[1] 8023  100
```

5.1.4) Compute Driver-Based Tumour—Model Similarity

We compute Spearman correlations between patient tumours and models using only driver—associated CpGs, mirroring the DEA-restricted approach from Tutorial 4, resulting in a driver-restricted tumour—model similarity table:

Implementation:

```
beta_cells_driver[]    <- lapply(beta_cells_driver, as.numeric)
 beta_patients_driver[] <- lapply(beta_patients_driver, as.numeric)


 driver_cor_results <- expand.grid(
    CellLine = colnames(beta_cells_driver),
    Patient  = colnames(beta_patients_driver),
    KEEP.OUT.ATTRS = FALSE
 )
 driver_cor_results$Correlation <- mapply(
    function(cl, pt) {
        suppressWarnings(
            cor(
                beta_cells_driver[, cl],
                beta_patients_driver[, pt],
                method = "spearman",
                use = "pairwise.complete.obs"
            )
        )
    },

    driver_cor_results$CellLine,
    driver_cor_results$Patient
 )

head(driver_cor_results)
#                      CellLine             Patient Correlation
#1 GSM1669626_7878191160_R04C01 203219670039_R07C01   0.7865550
#2 GSM1669637_7878191160_R03C01 203219670039_R07C01   0.7554586
#3 GSM1669639_8221924127_R04C01 203219670039_R07C01   0.7972434
#4 GSM1669651_8221932016_R06C01 203219670039_R07C01   0.7945049
#5 GSM1669875_8221916156_R03C02 203219670039_R07C01   0.7929934
#6 GSM1669876_8221932016_R06C02 203219670039_R07C01   0.7982920
```

5.1.5) Annotate Correlations with Clinical and Model Metadata

Annotation enables stratified visualisation and biological interpretation linking driver—gene similarity patterns to clinical variables and mapping internal IDs to interpretable model names:

Implementation:

```
driver_cor_results_annotated <- driver_cor_results %>%
    left_join(
        clinical_patient_aligned %>%
            select(
                Sentrix_ID,
                HIV_status,
                HPV_Genotype,
                Age,
                BMI,
                Cancer_Stage
            ),
        by = c("Patient" = "Sentrix_ID")
    ) %>%
    left_join(
        cell_meta_aligned %>%
            select(Beta_Column, Cell_Symbol),
```

```
        by = c("CellLine" = "Beta_Column")
    ) %>%
    mutate(
        CellLine = ifelse(is.na(Cell_Symbol), CellLine, Cell_Symbol),
        Age = as.numeric(Age),
        BMI = as.numeric(BMI),
        Correlation = as.numeric(Correlation)
    ) %>%
    select(-Cell_Symbol)

 head(driver_cor_results_annotated)
#  CellLine             Patient Correlation HIV_status HPV_Genotype Age      BMI
#1     BOKU 203219670039_R07C01   0.7865550   Negative           NA  60 23.87511
#2   C-33-A 203219670039_R07C01   0.7554586   Negative           NA  60 23.87511
#3    C-4-I 203219670039_R07C01   0.7972434   Negative           NA  60 23.87511
#4   CAL-39 203219670039_R07C01   0.7945049   Negative           NA  60 23.87511
#5     HELA 203219670039_R07C01   0.7929934   Negative           NA  60 23.87511
#6   HELASF 203219670039_R07C01   0.7982920   Negative           NA  60 23.87511
#  Cancer_Stage
#1          III
#2          III
#3          III
#4          III
#5          III
#6          III
```

5.1.6) DEA-Restricted Driver—Gene Methylome Visualisations

This tutorial applies the DEA-restricted visualisation framework from Tutorial 4 to cancer driver gene—associated DMPs. By restricting analyses to CpGs mapped to validated cancer driver genes, we assess tumour—model concordance within oncogenic epigenetic programs. Heatmaps visualiise coherent driver—level methylation structure across patients, while multi—line similarity profiles reveal how individual models behave across the full patient cohort at the driver—gene level.

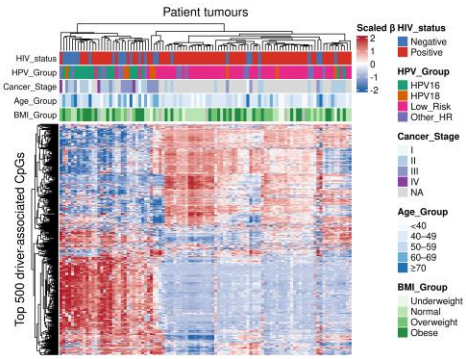
5.1.6.1) Heatmap of Top Driver—Gene DMPs

To assess whether driver—gene—associated methylation captures coherent epigenetic structure, we visualise the top differentially methylated CpGs mapped to cancer driver genes across patients. CpGs are scale per probe, and patients are annotated by clinical groups exactly as in Tutorial 4:

Implementation:

```
# Select top 500 driver-associated CpGs by variance
 driver_var <- apply(beta_patients_driver, 1, var, na.rm = TRUE
 top_driver_ids <- names(sort(driver_var, decreasing = TRUE))[1:500]
 beta_top_driver <- beta_patients_driver[top_driver_ids, , drop = FALSE]
 beta_top_driver_scaled <- t(scale(t(beta_top_driver)))
 stopifnot(
    all(colnames(beta_top_driver_scaled) %in% clinical_patient_aligned$Sentrix_ID)
 )
```

```
library(tibble)
library(dplyr)
patient_anno <- clinical_patient_aligned %>%
   filter(Sentrix_ID %in% colnames(beta_top_driver_scaled)) %>%
   distinct(Sentrix_ID, .keep_all = TRUE) %>%
   column_to_rownames("Sentrix_ID") %>%
   .[colnames(beta_top_driver_scaled), ]


#Driver-gene DMP heatmap (identical settings)
ha <- HeatmapAnnotation(
   df = patient_anno %>%
       select(
           HIV_status,
           HPV_Group,
           Cancer_Stage,
           Age_Group,
           BMI_Group
       ),
   col = anno_colours,
   annotation_name_side = "left",
   annotation_name_gp = grid::gpar(fontsize = 9)
)
tiff(
   "Heatmap_Top500_Driver_CpGs_Clinical_Annotations.tiff",
   width = 4200,
   height = 3200,
   res = 600,
   compression = "lzw"
)
Heatmap(
   beta_top_driver_scaled,
   name = "Scaled β",
   top_annotation = ha,
   show_row_names = FALSE,
   show_column_names = FALSE,
   cluster_rows = TRUE,
   cluster_columns = TRUE,
   row_title = "Top 500 driver-associated CpGs",
   column_title = "Patient tumours",
   col = colorRamp2(
       c(-2, 0, 2),
       c("#2166AC", "white", "#B2182B")
   )
)

dev.off()
```

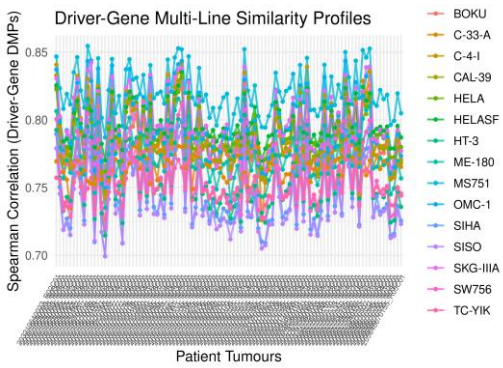


5.1.6.2) Multi—line DEA-Restricted Driver—Gene Similarity Profiles

Multi—line plots visualise Spearman correlations computed using driver gene DMPs, showing how each model behaves across all patients and identifying consistently concordant or patient—specific driver—level mapping.

Implementation:

```
driver_cor_results_annotated <- driver_cor_results_annotated %>%
  mutate(
    Patient = as.character(Patient),
    CellLine = as.character(CellLine),
    Correlation = as.numeric(Correlation)
  )
#Multi-line plot (identical to Tutorial 4)
tiff(
  "DEA_Restricted_DriverGene_MultiLine_Similarity.tiff",
  width = 3500, height = 2500, res = 600, compression = "lzw"
)

ggplot(driver_cor_results_annotated,
       aes(x = Patient, y = Correlation,
           group = CellLine, color = CellLine)) +
  geom_line(alpha = 0.8, linewidth = 0.7) +
  geom_point(size = 1) +
  labs(
    title = "DEA-Restricted Driver-Gene Multi-Line Similarity Profiles",
    x = "Patient Tumours",
    y = "Spearman Correlation (Driver-Gene DMPs)"
  ) +
  theme_minimal(base_size = 12) +
  theme(
    axis.text.x = element_text(angle = 60, hjust = 1, size = 6),
    legend.position = "right"
  )

dev.off()
```



5.2) Driver-Restricted ML-Based Tumour—Model Mapping

Driver-restricted methylation similarity provides a mechanistically informed representation of tumour—model concordance by focusing exclusively on CpGs linked to known cancer driver genes. This removes background methylation variation and emphasises oncogenic programs more directly related to tumour biology. However, as with DEA-restricted similarity, absolute Spearman correlations alone are insufficient for model prioritisation. Two tumours may show comparable absolute similarity to a model yet differ markedly in how exceptional that model is relative to all others tested for the same patient. Biological relevance therefore lies in within—patient relative concordance, motivating patient—normalised features and ML—based ranking. We integrate asolute driver—restricted similarity with relative ranking and deviation metrics in a Random Forest framework to derive ppatient—specific, data—driven tumour—model rankings.

Prerequisites

`driver_cor_results_annotated must contain:`

- `Patient`
- `CellLine`
- `Correlation (driver-restricted Spearman)`

5.2.1) Input Data Preparation

Each row represents a single patient—model pairing. Features are restricted to driver—based similarity metrics only, ensuring interpretability and direct biological relevance.

`Implementation:`

```
ml_features_driver <- driver_cor_results_annotated %>%
    select(Patient, CellLine, Driver_Cor = Correlation)
```

5.2.2) Within-Patient Normalisation (Relative Similarity)

Absolute driver-restricted correlations are not comparable across patients due to baseline epigenetic variability. We therefore normalise within each patient, capturing how exceptional each model is relative to all others for the same tumour. Two

complementary representations are used, Driver_Rank and Driver_Z. They respectively represent ordinal ranking of models per patient, and standardised deviation from the patient—specific mean:

Implementation:

```
ml_features_driver <- ml_features_driver %>%
    group_by(Patient) %>%
    mutate(
        Driver_Rank = rank(-Driver_Cor, ties.method = "average"),
        Driver_Z    = scale(Driver_Cor)
    ) %>%
    ungroup()
```

5.2.3) Definining Training Labels (Weak Supervision)

As no ground-truth optimal model exists, we use weak supervision. The top n driver—restricted models per patient (here n = 3) are treated as positive examples:

Implementation:

```
 ml_features_driver <- ml_features_driver %>%
    mutate(
        Label = ifelse(Driver_Rank <= 3, 1, 0)
    )
```

5.2.4) ML Model Selection

Random Forests are preferred because they handle correlated pedictors, capture non—linear, threshold—like ranking behaviour, are robust to separate without distributional assumptions:

Implementation:

```
library(caret)
 ml_features_driver <- ml_features_driver %>%
    mutate(
        Label = factor(
            ifelse(Label == 1, "Yes", "No"),
            levels = c("No", "Yes")
        )
    )

 set.seed(123)

 train_idx <- createDataPartition(
   ml_features_driver$Label,
   p = 0.7,
    list = FALSE
)
```

```
 train_data <- ml_features_driver[train_idx, ]
 test_data  <- ml_features_driver[-train_idx, ]

 rf_driver_model <- train(
    Label ~ Driver_Cor + Driver_Z + Driver_Rank,
    data = train_data,
    method = "rf",
    trControl = trainControl(
        method = "cv",
        number = 5,
        classProbs = TRUE,
        summaryFunction = twoClassSummary
    ),
    metric = "ROC",
    tuneLength = 5
 )
#note: only 2 unique complexity parameters in default grid. Truncating the grid to 2 .
```

5.2.5) ML Model Evaluation

5.2.5.1) Model-Derived Tumour—model Ranking

Performance is assessed using ranking—aware criteria rather than raw accuracy to distinguish top vs non—top models (ROC-AUC), investigate correctness of top—ranked selections, and stability of priorritised models across patients:

Implementation:

```
pred_probs_driver <- predict(
    rf_driver_model,
    test_data,
    type = "prob"
 )[, "Yes"]
```

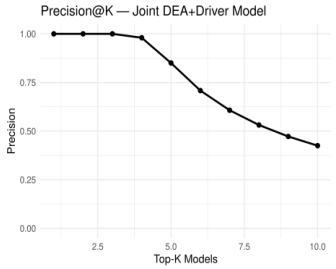
The precited probability of being a top model is used as a learned similarity score, integrating absolute driver similarity with relative within—patient performance.

Implementation:

```
ml_features_driver$ML_Score <- predict(
    rf_driver_model,
    ml_features_driver,
    type = "prob"
 )[, "Yes"]


 ml_driver_ranked <- ml_features_driver %>%
    group_by(Patient) %>%
    arrange(desc(ML_Score)) %>%
    mutate(ML_Rank = row_number()) %>%
    ungroup()
```

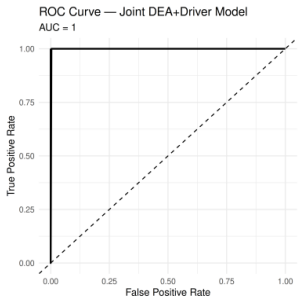
5.2.5.2) Precision@K for Joint Model Tumour—Model Ranking

For the joint model, Precision@K quantifies how well the integrated model prioritises top—performing cell lines per patient. By combining DEA, and driver features, this metric evaluates wether the joint ML framework improves the identification of high—fidelity models compared to using single—feature models.



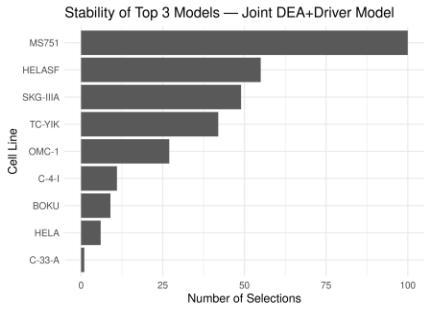
5.2.5.3) ROC Curve—Top vs Non—Top Model Classification (Joint Model)

The ROC curve shows the classification ability of the joint model to distinguish top vs non—top models across all patients. It allows comparison of discriminative power with single—feature models. A higher AUC indicates better integration of multiple biological signals.



5.2.5.4) Stability of Top—K models Across Patients (Joint Model)

This visualisation meeasures how consistent the joint model's top—K predictions are across patients. High stability indicates that certain ccell lines are robustly identified as the best match when integrating DEA, and Driver features. This is useful for prioritising models for experimental validation.

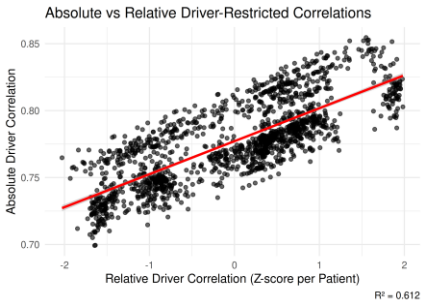


5.2.5.5) Absolute vs Relative Driver Similarity

To empirically demonstrate that relative concordance caries additional signal, we regress absolute driver—restricted correlations against patient—specific Z—scores.

Implementation:

```
reg_data_driver <- ml_features_driver %>%
    select(Patient, CellLine, Driver_Cor, Driver_Z)


 reg_model_driver <- lm(Driver_Cor ~ Driver_Z, data = reg_data_driver)
 summary(reg_model_driver)
#Call:
#lm(formula = Driver_Cor ~ Driver_Z, data = reg_data_driver)
#
#Residuals:
#      Min        1Q    Median        3Q       Max
#-0.037395 -0.013653 -0.006011  0.018495  0.041511
#
#Coefficients:
#             Estimate Std. Error t value Pr(>|t|)
#(Intercept) 0.7771637  0.0004901 1585.84   <2e-16 ***
#Driver_Z    0.0246401  0.0005073   48.57   <2e-16 ***
#---
#Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

#Residual standard error: 0.01898 on 1498 degrees of freedom
#Multiple R-squared:  0.6117, Adjusted R-squared:  0.6114
#F-statistic:  2359 on 1 and 1498 DF,  p-value: < 2.2e-16
 tiff(
    "ML_Driver_Relative_vs_Absolute_Correlation.tiff",
    width = 3500,
    height = 2500,
    res = 600,
    compression = "lzw"
 )


 ggplot(reg_data_driver, aes(x = Driver_Z, y = Driver_Cor)) +
    geom_point(alpha = 0.6, size = 1.5) +
    geom_smooth(method = "lm", color = "red", se = TRUE) +
    theme_minimal(base_size = 12) +
    labs(
        title = "Absolute vs Relative Driver-Restricted Correlations",
        x = "Relative Driver Correlation (Z-score per Patient)",
        y = "Absolute Driver Correlation",
        caption = paste0(
            "R² = ",
            round(summary(reg_model_driver)$r.squared, 3)
        )
    )
#`geom_smooth()` using formula = 'y ~ x'


 dev.off()
```



Driver—restricted relative similarity explains a substantially larger fraction of the variance in absolute correlations ($R^2 = 0.612$) compared with DEA-restricted correlations ($R^2 = 0.211$). This indicates that within—patient normalisation is particularly informative when similarity is computed over cancer driver—associated CpGs, where methylation variation is more directly coupled to oncogenic processes rather than global or immune associated effects (CC-HIV-Pos vs CC-HIV-Neg). These results validate the ML—based tumour—model prioritisation framework and highlight driver—focused epigenetic programs as a strong substrate for model mapping. They further motivate integration with complemeentary feature layers, including tumour microenvironment (TME) signatures, pathway—level hallmarks, and driver—specific functional annotations, to refine patient—specific model selection.

6. TME-Restricted Tumour—BioModel Mapping

Tumours exist within a complex tumour microenvironment (TME) composed of immune, stromal, vascular, and cancerr—cell compartments. Bulk methylation profiles therefore encode both intrinsic tumour programs, and extrinsic microenvironmental signals. In this Tutorial, we restrict methylation analyses to CpGs associated with curated TME marker genes, allowing tumour—model similarity to be evaluated through a microenvironment—aware epigenetic lens, analogous to the DEA—and driver—restricted frameworks introduced earlier.

The following packages are required for this tutorial:

```
library(ComplexHeatmap)
library(circlize)
library(dplyr)
library(tibble)
```

6.1) TME-Restricted tumour—biomodel Mapping

6.1.1) Data Preparation & CpG Subsetting

6.1.1.1) TME Gene Sets

We use literature—curated markers reppresenting key TME compartments.

Implementation:

```
markers <- list(
   CAF = c("FAP","ACTA2","PDGFRB","PDPN","NT5E","COL1A1","COL1A2","THY1","VCAN"),
   Immune_Inflam = c("PTPRC","CD3D","CD3E","CD8A","CD68","CD14",
                     "IL1B","TNF","CXCL9","CXCL10"),
   Stemness = c("LGR5","ALDH1A1","PROM1","SOX2","NANOG","POU5F1","CD44"),
   Cancer_Epithelial = c("EPCAM","KRT8","KRT18","KRT19","CDH1","MUC1"),
   Endothelial = c("PECAM1","VWF","CDH5","KDR","FLT1","TEK"),
   Pericyte = c("RGS5","PDGFRB","MCAM","ANGPT1","CSPG4")
)

tme_genes <- unique(unlist(markers))
```

6.1.1.2) Map CpGs to TME Genes

Using the curated marker set, we identify CpGs mapped to TME markers from EPIC-array annotation. These CpGs will be extracted and used to create methylation matrices for both cell lines and patient tumours.

Implementation:

```
anno <- as.data.frame(
   getAnnotation(IlluminaHumanMethylationEPICanno.ilm10b4.hg19)
)
```

```
tme_cpgs <- anno %>%
   filter(UCSC_RefGene_Name %in% tme_genes) %>%
   select(
       CpG = Name,
       Gene = UCSC_RefGene_Name,
       chr,
       pos
   ) %>%
   distinct()
```

6.1.1.3) Preparing TME-Restricted β-Matrices

Both patient and cancer in—vitro model matrices are subset to CpGs annotated to the curated TME marker genes. Only CpGs present in both datasets are retained to ensure comparability.

Implementation:

```
beta_cell    <- beta_cell[common_cpgs, , drop = FALSE]
beta_patient <- beta_patient[common_cpgs, , drop = FALSE]

common_tme_cpgs <- intersect(
   tme_cpgs$CpG,
   intersect(
       rownames(beta_patient),
       rownames(beta_cell)
   )
)


beta_patients_tme <- beta_patient[common_tme_cpgs, , drop = FALSE]
beta_cells_tme    <- beta_cell[common_tme_cpgs, , drop = FALSE]


# Ensure numeric matrices
beta_cells_tme[]    <- lapply(beta_cells_tme, as.numeric)
beta_patients_tme[] <- lapply(beta_patients_tme, as.numeric)


# Confirm matched CpGs
stopifnot(identical(rownames(beta_cells_tme), rownames(beta_patients_tme)))
```

6.1.2) Computing TME—Restricted Tumour—Model Similarity

After preparing the TME methylation matrices, we calculate Spearman correlations between the TME methylation profiles of cervical cancer in—vitro models and patient tumours.

Implementation:

```
tme_cor_results <- expand.grid(
    CellLine = colnames(beta_cells_tme),
    Patient  = colnames(beta_patients_tme),
    KEEP.OUT.ATTRS = FALSE
)


tme_cor_results$Correlation <- mapply(
```

```
    function(cl, pt) {
        suppressWarnings(
            cor(
                beta_cells_tme[, cl],
                beta_patients_tme[, pt],
                method = "spearman",
                use = "pairwise.complete.obs"
            )
        )
    },
    tme_cor_results$CellLine,
    tme_cor_results$Patient
 )
 head(tme_cor_results)
#                       CellLine             Patient Correlation
#1 GSM1669626_7878191160_R04C01 203219670039_R07C01   0.5409294
#2 GSM1669637_7878191160_R03C01 203219670039_R07C01   0.4468717
#3 GSM1669639_8221924127_R04C01 203219670039_R07C01   0.5833474
#4 GSM1669651_8221932016_R06C01 203219670039_R07C01   0.6172007
#5 GSM1669875_8221916156_R03C02 203219670039_R07C01   0.6430923
#6 GSM1669876_8221932016_R06C02 203219670039_R07C01   0.6362162
```

6.1.3) Clinical and Model Annotation

To support interpretability and visualisation, correlation results are annotated with patient clinical metadata and cell—line identifiers, enforcing consistent naming and data types.

Implementation:

```
 tme_cor_results_annotated <- tme_cor_results %>%
    left_join(
        clinical_patient_aligned %>%
            select(
                Sentrix_ID,
                HIV_status,
                HPV_Genotype,
                Age,
                BMI,
                Cancer_Stage
            ),
        by = c("Patient" = "Sentrix_ID")
    ) %>%
    left_join(
        cell_meta_aligned %>%
            select(Beta_Column, Cell_Symbol),
        by = c("CellLine" = "Beta_Column")
    ) %>%
    mutate(
        CellLine = ifelse(is.na(Cell_Symbol), CellLine, Cell_Symbol),
        Age = as.numeric(Age),
        BMI = as.numeric(BMI),
        Correlation = as.numeric(Correlation)
    ) %>%
    select(-Cell_Symbol)


 head(tme_cor_results_annotated)
```

```
#  CellLine             Patient Correlation HIV_status HPV_Genotype Age      BMI
#1     BOKU 203219670039_R07C01   0.5409294   Negative           NA  60 23.87511
#2   C-33-A 203219670039_R07C01   0.4468717   Negative           NA  60 23.87511
#3    C-4-I 203219670039_R07C01   0.5833474   Negative           NA  60 23.87511
#4   CAL-39 203219670039_R07C01   0.6172007   Negative           NA  60 23.87511
#5     HELA 203219670039_R07C01   0.6430923   Negative           NA  60 23.87511
#6   HELASF 203219670039_R07C01   0.6362162   Negative           NA  60 23.87511
#  Cancer_Stage
#1          III
#2          III
#3          III
#4          III
#5          III
#6          III
```

6.1.4) Heatmap of Highly Variable TME-Associated CpGs

By selecting the top 500 TME—associated CpGs ranked by variance and visualising their scaled methylation profiles, this heatmap captures dominant TME-driven epigenetic patterns across patients.

Implementation:

```
## Select top 500 TME CpGs by variance
tme_var <- apply(beta_patients_tme, 1, var, na.rm = TRUE)
top_tme_ids <- names(sort(tme_var, decreasing = TRUE))[1:500]


beta_top_tme <- beta_patients_tme[top_tme_ids, , drop = FALSE]
beta_top_tme_scaled <- t(scale(t(beta_top_tme)))


## Remove CpGs with NA after scaling
beta_top_tme_scaled <- beta_top_tme_scaled[
   rowSums(is.na(beta_top_tme_scaled)) == 0,
]
```

Overlaying clinical and viral annotations (HIV status, HPV group, cancer stage, age, BMI) enables assessment of whether TME-linked methylation structure aligns with known clinical covariates, supporting interpretability, and hypothesis generation.

Implementation:

```
## Patient annotation (ordered to match heatmap)
patient_anno <- clinical_patient_aligned %>%
   filter(Sentrix_ID %in% colnames(beta_top_tme_scaled)) %>%
   distinct(Sentrix_ID, .keep_all = TRUE) %>%
   column_to_rownames("Sentrix_ID") %>%
   .[colnames(beta_top_tme_scaled), ] %>%
   mutate(
       Age = as.numeric(Age),
       BMI = as.numeric(BMI),


       Age_Group = cut(
           Age,
           breaks = c(0, 40, 50, 60, 70, Inf),
           labels = c("<40", "40-49", "50-59", "60-69", "≥70"),
           right = FALSE
```

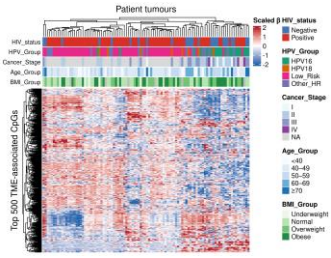
```
        ),

        BMI_Group = cut(
            BMI,
            breaks = c(0, 18.5, 25, 30, Inf),
            labels = c("Underweight", "Normal", "Overweight", "Obese"),
            right = FALSE
        ),

        HPV_Genotype = as.character(HPV_Genotype),
        HPV_Group = case_when(
            HPV_Genotype == "16" ~ "HPV16",
            HPV_Genotype == "18" ~ "HPV18",
            HPV_Genotype %in% c(
                "31","33","35","39","45","51","52","56","58","59","68","73","82"
            ) ~ "Other_HR",
            is.na(HPV_Genotype) ~ "Unknown",
            TRUE ~ "Low_Risk"
        ),

        Cancer_Stage = ifelse(is.na(Cancer_Stage), "NA", Cancer_Stage)
    )


ha <- HeatmapAnnotation(
    df = patient_anno %>%
        select(HIV_status, HPV_Group, Cancer_Stage, Age_Group, BMI_Group),
    col = anno_colours,
    annotation_name_side = "left",
    annotation_name_gp = grid::gpar(fontsize = 9)
)
```

Heatmap of top 500 variable TME—associated CpGs with clinical annotations in publication—ready format (TIFF, 600dpi, LZW compression).

Implementation:

```
tiff(
    "Heatmap_Top500_TME_CpGs_Clinical_Annotations.tiff",
    width = 4200, height = 3200, res = 600, compression = "lzw"
)


Heatmap(
    beta_top_tme_scaled,
    name = "Scaled β",
    top_annotation = ha,
    show_row_names = FALSE,
    show_column_names = FALSE,
    cluster_rows = TRUE,
    cluster_columns = TRUE,
    row_title = "Top 500 TME-associated CpGs",
    column_title = "Patient tumours",
    col = colorRamp2(
        c(-2, 0, 2),
        c("#2166AC", "white", "#B2182B")
    )
)
```

```
dev.off()
```



6.1.5) TME-Restricted Tumour—Model Similarity Profiles

While intrinsic differential expression or driver—based similarity highlights tumour—cell autonomous programmes, TME—restricted correlations capture contextual similarity driven by immune infiltration, stromal activation, vascular support, and epithelial composition. Line—based visualisation preserves within—model trends while allowing cross-model comparison. DEA-restricted TME multi—line similarity profiles in publish ready format.

Implementation:

```
library(ggplot2)
tiff(
   "DEA_Restricted_TME_MultiLine_Similarity.tiff",
   width = 3500,
   height = 2500,
   res = 600,
   compression = "lzw"
)


ggplot(
   tme_cor_results_annotated,
   aes(
       x = Patient,
       y = Correlation,
       group = CellLine,
       color = CellLine
   )
) +
   geom_line(alpha = 0.8, linewidth = 0.7) +
   geom_point(size = 1) +
   labs(
       title = "DEA-Restricted TME Multi-Line Similarity Profiles",
       x = "Patient Tumours",
       y = "Spearman Correlation (TME-associated CpGs)"
   ) +
   theme_minimal(base_size = 12) +
   theme(
       axis.text.x = element_text(angle = 60, hjust = 1, size = 6),
       legend.position = "right"
   )
```

```
dev.off()
```

6.2) TME—Restricted ML—Based Tumour—Model Mapping

TME-restricted methylation similarity captures tumour—model concordance driven by microenvironmental programs, including immune infiltration, stromal activation, vascular support, and stem—like states. While absolute Spearman correlations quantify overall similarity, they do not capture how exceptional a model is relative to others for the same tumour. We therefore integrate absolute TME—restricted similarity with within—patient relative ranking and deviation metrics in Random Forest framework to derive patient—specific, data—driven tumour—model prioritisation:

6.2.1) Input Feature Construction

We restrict features exclusively to TME—derived similarity metrics to preserve interpretability and ensure the learned signal reflects microenvironmental concorrdance rather than confounding molecular layers:

Implementation:

```
ml_features_tme <- tme_cor_results_annotated %>%
    select(Patient, CellLine, TME_Cor = Correlation)
```

6.2.2) Within—Patient Normalisation (Relative Similarity)

Absolute TME similarity values are not comparable across patients due to baseline differences in immune and stromal composition. We therefore normalise within each patient, capturing how exceptional a model is relative to all others tested for the same tumour. Two complementary representations are used ordinal dominance among models (TME_Rank), and standardised deviation from the patient—specific mean.

Implementation:

```
ml_features_tme <- ml_features_tme %>%
   group_by(Patient) %>%
   mutate(
       TME_Rank = rank(-TME_Cor, ties.method = "average"),
       TME_Z    = scale(TME_Cor)
   ) %>%
   ungroup()
```

6.2.3) Weak Supervision Label Definition

No ground-truth “best” tumour model exists. We therefore apply weak supervision, where the top—performing TME—restricted models per patient (n=3) are treated as positive examples. This converts ranking into a learnable classification problem without external bias.

Implementation:

```
ml_features_tme <- ml_features_tme %>%
    mutate(
        Label = factor(
            ifelse(TME_Rank <= 3, "Yes", "No"),
            levels = c("No", "Yes")
        )
    )
```

6.2.4) ML Model Selection

Random Forests are selected because they handle correlated predictors (TME_Cor, TME_Z, TME_Rank), and capture non-linear, threshold—like ranking behaviour. Moreover, RF models are robust to weakly supervised labels, and do not assume linear separability or normality. Logistic regression fails in this context due to rank—derived label leakage and quasi—separation.

Implementation:

```
library(caret)
#Loading required package: lattice
#Attaching package: ‘caret’
#The following object is masked from ‘package:generics’
#    train


 set.seed(123)


 train_idx <- createDataPartition(
    ml_features_tme$Label, p = 0.7, list = FALSE
 )


 train_data <- ml_features_tme[train_idx, ]
 test_data  <- ml_features_tme[-train_idx, ]


 rf_tme_model <- train(
    Label ~ TME_Cor + TME_Z + TME_Rank,
    data = train_data,
    method = "rf",
    trControl = trainControl(
        method = "cv",
        number = 5,
        classProbs = TRUE,
        summaryFunction = twoClassSummary
    ),
    metric = "ROC",
```

```
    tuneLength = 5
 )
#note: only 2 unique complexity parameters in default grid. Truncating the grid to 2.
```

6.2.5) Model-Derived Tumour—Model Ranking

The predicted probability of belonging to the top—ranked class serves as a learned similarity score, integrating absolute TME similarity with relative within—patient dominance. This replaces heuristic correlation cut-offs with patient—specific, data—driven prioritisation.

Implementation:

```
ml_features_tme$ML_Score <- predict(
  rf_tme_model,
  ml_features_tme,
  type = "prob"
)[, "Yes"]
ml_tme_ranked <- ml_features_tme %>%
  group_by(Patient) %>%
  arrange(desc(ML_Score)) %>%
  mutate(ML_Rank = row_number()) %>%
  ungroup()
```

6.3) Absolute vs Relative TME—Restricted Similarity

To empirically demonstrate that within—patient relative TME similarity captures additional biological signal beyond raw absolute correlations, we regress absolute TME—restricted Spearman correlations against patient—specific Z—scores. This analysis tests whether relative scaling explains systematic variation in absolute tumour—model similarity when TME programs are considered.

Prerequisites

ml_features_tme must contain:

- Patient
- CellLine
- TME_Cor (absolute TME-restricted Spearman)
- TME_Z (within-patient Z-score)

Regression Model

The slope quantifies how strongly relative TME deviation predicts absolute similarity. The R-Squared measures how much inter—model variation is explained by within—patient normalisation alone.

Implementation:

```
 reg_data_tme <- ml_features_tme %>%
    select(Patient, CellLine, TME_Cor, TME_Z)
```

```r
reg_model_tme <- lm(TME_Cor ~ TME_Z, data = reg_data_tme)
summary(reg_model_tme)
```

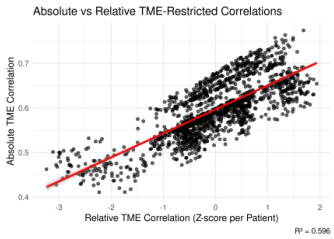
```
#Call:
#lm(formula = TME_Cor ~ TME_Z, data = reg_data_tme)

#Residuals:
#      Min        1Q    Median        3Q       Max
#-0.091751 -0.029294 -0.006811  0.040699  0.099309

#Coefficients:
#            Estimate Std. Error t value Pr(>|t|)
#(Intercept) 0.596805   0.001107  538.95   <2e-16 ***
#TME_Z       0.053842   0.001146   46.97   <2e-16 ***
#---
#Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

#Residual standard error: 0.04289 on 1498 degrees of freedom
#Multiple R-squared:  0.5956, Adjusted R-squared:  0.5954
#F-statistic:  2207 on 1 and 1498 DF,  p-value: < 2.2e-16
```

```r
tiff(
   "ML_TME_Relative_vs_Absolute_Correlation.tiff",
   width = 3500,
   height = 2500,
   res = 600,
   compression = "lzw"
)
```

```r
ggplot(reg_data_tme, aes(x = TME_Z, y = TME_Cor)) +
   geom_point(alpha = 0.6, size = 1.5) +
   geom_smooth(method = "lm", color = "red", se = TRUE) +
   theme_minimal(base_size = 12) +
   labs(
       title = "Absolute vs Relative TME-Restricted Correlations",
       x = "Relative TME Correlation (Z-score per Patient)",
       y = "Absolute TME Correlation",
       caption = paste0(
           "R² = ",
           round(summary(reg_model_tme)$r.squared, 3)
       )
   )
```

```
#`geom_smooth()` using formula = 'y ~ x'
```

```r
dev.off()
```



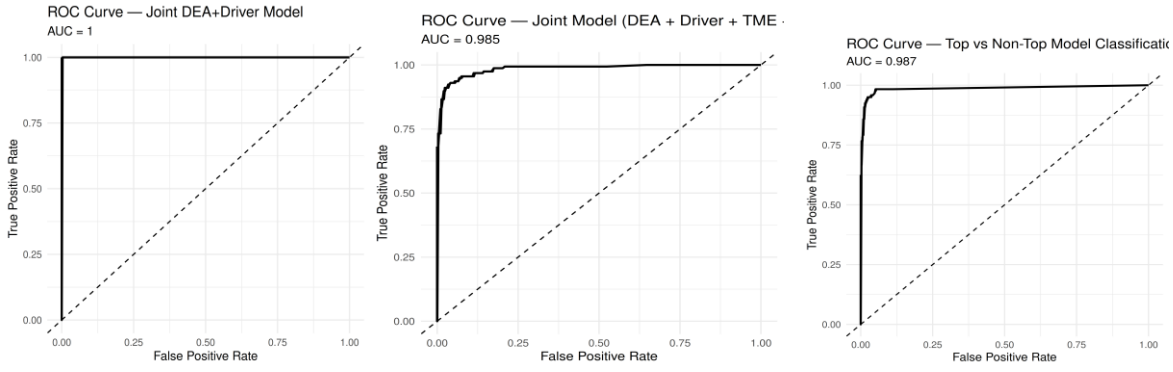
Regression analysis of absolute TME-restricted Spearman correlations against within—patient relative Z-scores yielded R-squared of 0.596, indicating that relative (patient—normalised) similarity explains a substantial proportion of the variance in absolute tumour—model concordance. Confirming that TME—focused relative concordance is highly informative, supporting ML-based tumour—model prioritisation using microenvironment—restricted epigenetic programs. Residual unexplained variance suggests complementary contributions from additional axes such as driver—gene programs and cancer hallmark—level signatures, motivating multi—feature integration.

6.4) Model Performance Evaluation of Tumour—Model Mapping Framework

This section evaluates the perfomance of the weakly supervised machine learning framework developed to prioritise cervical cancer models that best represent individual patient tumous. Model performance was assessed across two biologically motivated feature spaces; Joint Model integrating TME methylation, cancer driver gene—associated methylation, and differential methylation probes (DMPs); driver gene—associated methylation, and TME methylation only.

6.4.1) ROC and AUC—Discriminative Power

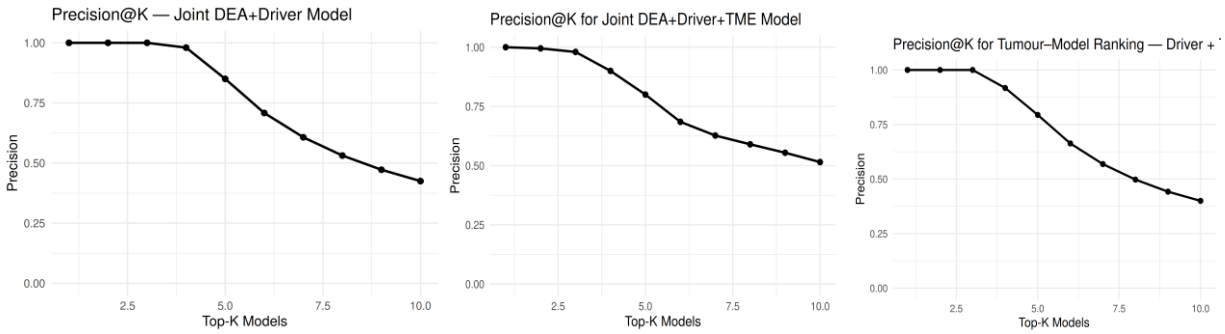
To evaluate the classifier's ability to separate true high—confidence tumour—model mapping from non—representative models. The extremely high AUC values confirm that the classifier very effectively distinguishes biologically concordant vs discordant models, validating the predictive power of the methylation—derived features. These implies tumour—intrinsic methylation and microenvirnmental methylation carry the dominant biological information. DEA/DMP contribute refinement and stability, but the framework is fundamentally driven by meaningful tumour biology rather than noise.



6.4.2) Precision@K—Reliability of Ranking

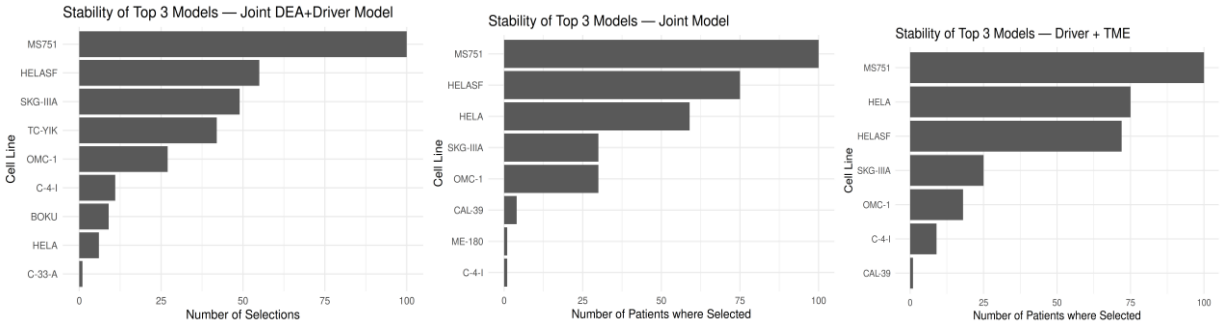
To evaluate whether the highest—ranked model predictions truly correspond to biologically concordant maps. Precision@K quantifies the proportion of correctly identified maps among the Top—K ranked models. Demonstrating that the rankings is

meaninful i.e., the top ranked models are almost always true maps, making the approach useful for recommending a small shortlist of representative models per patient.



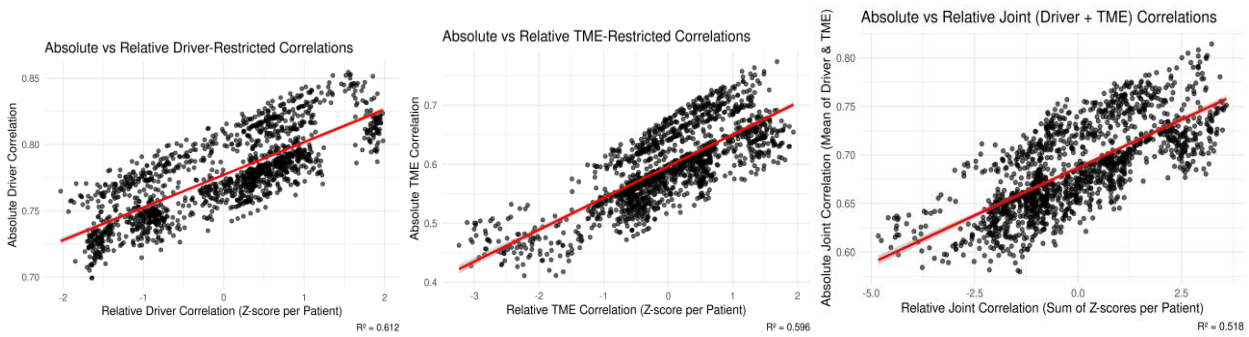
6.4.3) Stability of Predicted Tumour—Model Maps

To test whether the framework consistently prioritises similar models across patients rather than producing unstable or stochastic predictions. We see a strong recurrence of specific cervical cancer models across patients, indicating that the framework yields biologically stable and reproducible tumour—model maps rather than stochastic behaviour. These dominant models correspond to well—characterised HPV—driven cervical cancer models with transcriptional activity, viral integration, and proliferative signals reflective of patient tumours. This confirms that the framework is not random, tumour methylation structure aligns with expected biological models, and the system preferentially identifies clinically meaningful surrogates.



6.4.4) Absolute vs Relative Similarity

Across Driver, TME and Joint correlation frameworks, we observed strong linear associations between absolute similarity estimates and patient—standardised relativee simiilarity scores (Driver $R^2 = 0.612$; TME $R^2 = 0.596$; Joint $R^2 = 0.518$). This demostrates that relative Z—scored patient correlaton measures reliably preserve the biological structue of the absoluute simiilarity landscape, validating their use for downstream statification and model—mapping analyses.



The integration of microenvironmental and tumour—intrinsic methylation signals enable accurate, stable, and highly discriminative mapping of patient tumours to representative cervical cancer cell—line models. This framework provides a principled approach for rational model selection, supporting translational application in experimental modelling, drug discovery, and mechanistic studies.

7. TME-Restricted Tumour Stratification Using Patient—Wide Methylation Similarity

Tumour evolution is strongly influenced by tumour microenvironment (TME), including immune infiltration, stromal activation, vascularisation, and epithelial state. Rather than analysing individual CpGs or genes in isolation, we leverage patient—wide similarity profiles across all cell—line models, restricted to TME-associated CpGs curated from the literature. By clustering patients based on their global TME—restricted methylation similarity patterns, we aim to identify epigenetically coherent TME endotypes that transcend individual loci and capture system—level tumour states. This approach enables unsupervised discovery of biologically meaningful tumour communities and provides a foundation for functional interpretation.

The following R packages are required for this tutorial:

```
suppressPackageStartupMessages({
   library(dplyr)
   library(tidyr)
   library(tibble)
   library(ggplot2)
   library(ggrepel)
   library(ComplexHeatmap)
   library(circlize)
   library(Rphenograph)
   library(uwot)
})
```

7.1) TME-Restricted Methylome Clusters

7.1.1) Preparation of TME-Restricted Correlation Matrix

Clustering requires a matrix representation where each row corresponds to a patient and each column captures that patient's similarity profile across all cell—line models. Here, similarity is defined by Spearman correlations computed using only TME—associated CpGs (Tutorial 6). This matrix reflects how each patient relates to the full panel of models, providing a rich, multivariate representation of TME—driven epigenetic states. This step ensures a complete, noise-free similarity matrix suitable for graph-based clustering.

Implementation:

```
tme_cor_long <- tme_cor_results_annotated %>%
   select(
       Patient,
       CellLine,
       Correlation
   ) %>%
```

```
    mutate(
        Patient = as.character(Patient),
        CellLine = as.character(CellLine),
        Correlation = as.numeric(Correlation)
    )

 tme_mat <- tme_cor_long %>%
    pivot_wider(
        names_from  = CellLine,
        values_from = Correlation
    ) %>%
    column_to_rownames("Patient") %>%
    as.matrix()

# Remove cell lines or patients with missing values
 tme_mat <- tme_mat[
    rowSums(is.na(tme_mat)) == 0,
    colSums(is.na(tme_mat)) == 0
 ]
```

7.1.2) Rphenograph Clustering of TME Similarity Profiles

Rphenograph performs graph-based community detection using shared nearest neighbours, making it particularly well—suited for high—dimensional biological data. By clustering patients based on their TME similarity profiles, we identify tumour communities that shar comparable microenvironmental methylation programs. A neighbourhood size of k = 30 balances local sensitivity with global stability and has been shown to yield robust clustering in epigenomic data.

Implementation:

```
 set.seed(123)

 rph <- Rphenograph(tme_mat, k = 30)
#Run Rphenograph starts:
#  -Input data of 100 rows and 15 columns
#  -k is set to 30
#  Finding nearest neighbors...DONE ~ 0.012 s
#  Compute jaccard coefficient between nearest-neighbor sets...DONE ~ 0.044 s
#  Build undirected graph from the weighted links...DONE ~ 0.047 s
#  Run louvain clustering on the graph ...DONE ~ 0.024 s
#Run Rphenograph DONE, totally takes 0.127000000004045s.
#  Return a community class
#  -Modularity value: 0.5731933
#  -Number of clusters: 4
 clusters <- factor(membership(rph[[2]]))

 modularity_value <- rph[[2]]$modularity
 print(modularity_value)
#[1] 0.5731933
```

The observed Louvain modularity (±0.57) indicates strong community structure, supporting the presence of biologically meaningful TME—driven tumour subtypes.

7.1.3) Integration of Cluster Assignments with Clinical Metadata

To enable downstream interpretation, each patient is annotated with their TME cluster assignment alongside existing clinical metadata. This allows assessment of whether TME—defined clusters align with known clinical or pathological variables.

Implementation:

```
patient_meta <- data.frame(
   Patient = rownames(tme_mat),
   Cluster = clusters
) %>%
   left_join(
       clinical_patient_aligned,
       by = c("Patient" = "Sentrix_ID")
   )
```

7.1.4) UMAP Embedding of TME—Restricted Similarity Space

While clustering opeate in high—dimensional space, UMAP provides an intuitive low-dimensional embedding that preserves local neighbourhood structure. This allows visualisation of patient relationships and cluster separation driven by TME—restricted methylation patterns.

Implementation:

```
set.seed(123)


umap_res <- umap(
   tme_mat,
   n_neighbors = 30,
   min_dist = 0.3,
   metric = "euclidean"
)


umap_df <- as.data.frame(umap_res) %>%
   rename(UMAP1 = V1, UMAP2 = V2) %>%
   mutate(Patient = rownames(tme_mat)) %>%
   left_join(patient_meta, by = "Patient")
```

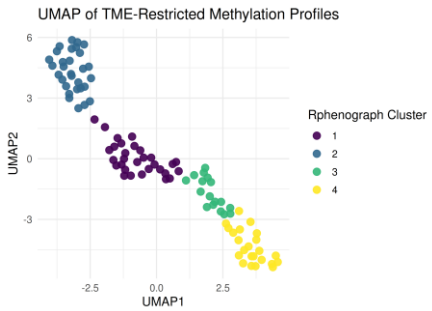
7.1.5) Visualisation of TME—driven Tumour Communities

UMAP projections coloured by Rphenograph clusters reveal TME—defined tumour communities, highlighting patient groups with shared microenvironmental epigenetic states. Compared to raw correlation plots, this representation reduces noise and emphasises global structure.

Implementation:

```
tiff(
   "UMAP_TME_Rphenograph_Clusters.tiff",
   width = 3500,
   height = 2500,
   res = 600,
```

```
   compression = "lzw"
)

ggplot(umap_df, aes(UMAP1, UMAP2, color = Cluster)) +
   geom_point(size = 3, alpha = 0.9) +
   theme_minimal(base_size = 12) +
   labs(
       title = "UMAP of TME-Restricted Methylation Profiles",
       color = "Rphenograph Cluster"
   ) +
   scale_color_viridis_d()

dev.off()
```



TME—Restricted methylation profiles reveal structured tumour communities driven by microenvironmental programs. Rphenograph clustering of patient—model similarity profiles identifies coherent tumour groups with strong modularity, while UMAP embeddings demonstrate clear low—dimensional separation. These patterns indicate that TME—associated CpGs capture biologically meaningful heterogeneity beyond tumour—intrinsic signals, motivating their integration into ML—based tumour—model prioritisation frameworks.

7.2) Model Performance Evaluation of Feature—Restricted Stratification

Traitional clinical stratification (HIV status, HPV genotype, staging) collapses divese tumours into coarse categories and fails to capture the functional molecular alignment between patient tumours and experimental models. To demostrate the benefit of a feature—restricted precision framework, we evaluatedd whether clustering based purely on model—tumour similarity scores (DEA, Driver and TME correlations) form biologically stable data—driven tumour states.

```
library(dplyr)
library(tidyr)
library(tibble)
```

7.2.1) Analytical Framework

For each patient, model alignment strengths were summaised into a joint correlation matrix. All incomplete rows/columns were removed to ensure a fully observed similarity

of space. This creates an interpretable patient embedding defined entirely by tumour—model similarity, free from clinical bias.

Implementation:

```
# Create long format for Joint correlations
 joint_cor_long <- ml_features_joint_clean %>%
    select(Patient, CellLine, DEA_Cor, Driver_Cor, TME_Cor) %>%
    mutate(
        Joint_Cor = rowMeans(cbind(DEA_Cor, Driver_Cor, TME_Cor), na.rm = TRUE)
    ) %>%
    select(Patient, CellLine, Joint_Cor) %>%
    mutate(
        Patient = as.character(Patient),
        CellLine = as.character(CellLine),
        Joint_Cor = as.numeric(Joint_Cor)
    )


 # Pivot to wide matrix: patients × cell lines
 joint_mat <- joint_cor_long %>%
    pivot_wider(
        names_from = CellLine,
        values_from = Joint_Cor
    ) %>%
    column_to_rownames("Patient") %>%
    as.matrix()


 # Remove rows/columns with NA
 joint_mat <- joint_mat[
    rowSums(is.na(joint_mat)) == 0,
    colSums(is.na(joint_mat)) == 0
 ]
```

7.2.2) Graph—Based Community Detection (Rphenograph)

Phenograph builds a k—nearest neighbour graph on patients, then applies Louvain community detection to identify molecular communities. Applying Rphenograph (k=30) to the joint matrix produced three robust patient communities with a similarly strong modularity score (±0.57), demostating well—defined clustering stucture. This indicates that integrating DEA, Driver, and TME layers successfully compresses tumour—model similarity information into stable biological communities.

Implementation:

```
library(Rphenograph)
 set.seed(123)


 rph_joint <- Rphenograph(joint_mat, k = 30)
#Run Rphenograph starts:
#  -Input data of 100 rows and 15 columns
#  -k is set to 30
#  Finding nearest neighbors...DONE ~ 0.007 s
#  Compute jaccard coefficient between nearest-neighbor sets...DONE ~ 0.042 s
#  Build undirected graph from the weighted links...DONE ~ 0.04 s
#  Run louvain clustering on the graph ...DONE ~ 0.021 s
```

```
#Run Rphenograph DONE, totally takes 0.109999999996944s.
#  Return a community class
#  -Modularity value: 0.5746973
#  -Number of clusters: 3>
 # Extract cluster memberships
 clusters_joint <- factor(membership(rph_joint[[2]]))


 # Modularity
 modularity_value <- rph_joint[[2]]$modularity
 print(modularity_value)
#[1] 0.5663016 0.5746973
```

7.2.3) Integration of Cluster Assignment with Clinical Metadata

To enable downstream interpretation, each patient is annotated with their integrated feature model cluster assignment alongside existing clinical metadata. This allows assessment of whether feature—defined clusters align with known clinical or pathological variables.

Implementation:

```
 # Assume you have a clinical patient table: clinical_patient_aligned
 patient_meta_joint <- data.frame(
    Patient = rownames(joint_mat),
    Cluster = clusters_joint
 ) %>%
    left_join(
        clinical_patient_aligned,
        by = c("Patient" = "Sentrix_ID")  # adjust column name if needed
    )
```

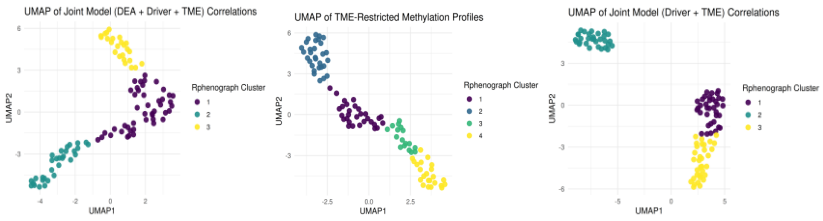
7.2.4) Low—Dimensional Projections (UMAP)

While clustering operates in high—ddimensional space, UMAP provides an intuitive low—ddimensional embedding that preserves local neighbourhood structure. This allows visualisation of patieent relationships and cluster separation driven by model methylation patterns.

Implementation:

```
 library(uwot)
 set.seed(123)


 umap_res_joint <- umap(
    joint_mat,
    n_neighbors = 30,
    min_dist = 0.3,
    metric = "euclidean"
 )


 umap_df_joint <- as.data.frame(umap_res_joint)
 colnames(umap_df_joint) <- c("UMAP1", "UMAP2")   # assign names explicitly
```

```
 umap_df_joint <- umap_df_joint %>%
    mutate(Patient = rownames(joint_mat)) %>%
    left_join(patient_meta_joint, by = "Patient")

 library(ggplot2)
 library(viridis)
#Loading required package: viridisLite

 tiff(
    "UMAP_Joint_Rphenograph_Clusters.tiff",
    width = 3500,
    height = 2500,
    res = 600,
    compression = "lzw"
 )

 ggplot(umap_df_joint, aes(UMAP1, UMAP2, color = Cluster)) +
    geom_point(size = 3, alpha = 0.9) +
    theme_minimal(base_size = 12) +
    labs(
        title = "UMAP of Joint Model (DEA + Driver + TME) Correlations",
        color = "Rphenograph Cluster"
    ) +
    scale_color_viridis_d()

 dev.off()
```



8. Clinical Stratification of Methylome Clusters

Tutorial 7 evaluates whether the four methylome clusters identified in Tutorial 6 correspond to clinical or virological features—including HIV status, HPV genotype, and diagnostic category (CIN vs cancer). Using t-SNE and UMAP embedding applied to the TME-derived methylation matrix, we project samples in 2D space and annotate them with key clinical labels. The goal is to determine whether methylation-defined endotypes correspond to clinical variables or whether they represent biology-driven tumour—intrinsic states. In this tutorial we will compute t-SNE and UMA embeddings from the TME methylation correlation matrix, annotate embedding with HIV status (HIV-Pos vs HIV-Neg), HPV genotype (distinct subtypes) and diagnostic class (CIN vs CC), and Rphenograph cluster (from Tutorial 7).

The following R packages are required for this tutorial;

```
library(Rtsne)
library(uwot)
library(ggplot2)
library(ggrepel)
library(dplyr)
```

8.1) Phenotypic Stratification of TME_Only Model Methylome Clusters

Data Requirement

From tutorial 7, we already have patient versus cell-line TME similarity matrix, and metadata including Cluster, HIV, HPV genotype and DiagnosticGroup;

Implementation:

```
set.seed(123)

umap_res <- umap(
   tme_mat,
   n_neighbors = 30,
   min_dist = 0.3,
   metric = "euclidean"
)


umap_df <- as.data.frame(umap_res) %>%
   rename(UMAP1 = V1, UMAP2 = V2) %>%
   mutate(Patient = rownames(tme_mat)) %>%
   left_join(patient_meta, by = "Patient")
# Long format for TME correlations
tme_cor_long <- tme_cor_results_annotated %>%
   select(Patient, CellLine, Correlation) %>%
   mutate(
       Patient = as.character(Patient),
       CellLine = as.character(CellLine),
       Correlation = as.numeric(Correlation)
```

```
    )

 # Pivot to wide matrix for clustering
 tme_mat <- tme_cor_long %>%
    pivot_wider(names_from = CellLine, values_from = Correlation) %>%
    column_to_rownames("Patient") %>%
    as.matrix()

 # Remove rows/columns with missing values
 tme_mat <- tme_mat[rowSums(is.na(tme_mat)) == 0,
                    colSums(is.na(tme_mat)) == 0]


 set.seed(123)
 rph <- Rphenograph(tme_mat, k = 30)
#Run Rphenograph starts:
#  -Input data of 100 rows and 15 columns
#  -k is set to 30
#  Finding nearest neighbors...DONE ~ 0.002 s
#  Compute jaccard coefficient between nearest-neighbor sets...DONE ~ 0.043 s
#  Build undirected graph from the weighted links...DONE ~ 0.016 s
#  Run louvain clustering on the graph ...DONE ~ 0.006 s
#Run Rphenograph DONE, totally takes 0.0669999999972788s.
#  Return a community class
#  -Modularity value: 0.5731933
#  -Number of clusters: 4
 clusters <- factor(membership(rph[[2]]))
 modularity_value <- rph[[2]]$modularity
 print(modularity_value)  # e.g., 0.5731933
#[1] 0.5731933
```

We begin by computing t-SNE and UMAP embeddings from the TME methylation correlation matrix. We then respectively, annotate embeddings with HIV status, HPV genotype, annd diagnostic class (CIN vs CC) and Rphenograph cluster (from Tutorial 7).

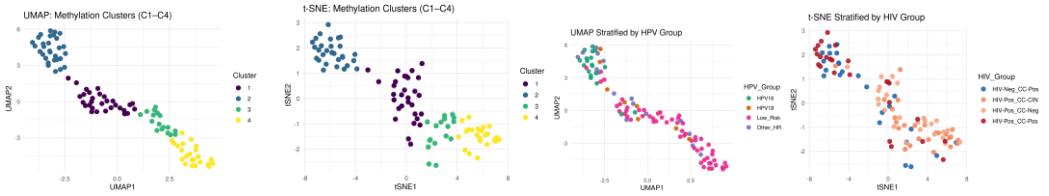
Implementation:

```
# t-SNE embedding
 set.seed(123)
 tsne_res <- Rtsne(tme_mat, perplexity = 30, max_iter = 1000, check_duplicates = FALSE)
 tsne_df <- data.frame(tSNE1 = tsne_res$Y[,1], tSNE2 = tsne_res$Y[,2], Patient =
rownames(tme_mat)) %>%
    left_join(patient_meta, by = "Patient")


 # UMAP embedding
 set.seed(123)
 umap_res <- umap(tme_mat, n_neighbors = 30, min_dist = 0.3)
 umap_df <- data.frame(UMAP1 = umap_res[,1], UMAP2 = umap_res[,2], Patient =
rownames(tme_mat)) %>%
    left_join(patient_meta, by = "Patient")
```

Next, we visualise cluster structure and evaluate clinical overlaps.

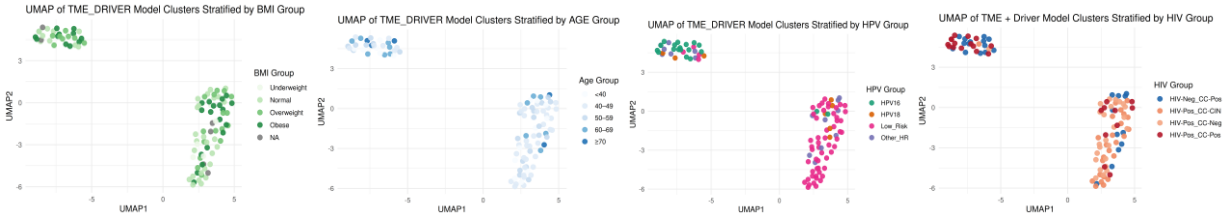
Implementation:

```
#Plot 1 – t-SNE Colored by Methylation Cluster

 tiff("tSNE_Rphenograph_Clusters.tiff",
     width = tiff_width, height = tiff_height,
     res = tiff_res, compression = tiff_compression)


 ggplot(tsne_df, aes(tSNE1, tSNE2, color = Cluster)) +
    geom_point(size = 3) +
    labs(title = "t-SNE: Methylation Clusters (C1-C4)") +
    theme_minimal(base_size = 12) +
    scale_color_viridis_d()


 dev.off()

#Plot 2 – UMAP Colored by Methylation Cluster

 tiff("UMAP_Rphenograph_Clusters.tiff",
     width = tiff_width, height = tiff_height,
     res = tiff_res, compression = tiff_compression)


 ggplot(umap_df, aes(UMAP1, UMAP2, color = Cluster)) +
    geom_point(size = 3) +
    labs(title = "UMAP: Methylation Clusters (C1-C4)") +
    theme_minimal(base_size = 12) +
    scale_color_viridis_d()


 dev.off()

#Plot 3 – UMAP Stratified by HIV Status

 tiff("UMAP_HIV_Group.tiff",
     width = tiff_width, height = tiff_height,
     res = tiff_res, compression = tiff_compression)


 ggplot(umap_df, aes(UMAP1, UMAP2, color = HIV_Group.x)) +
    geom_point(size = 3, alpha = 0.9) +
    labs(title = "UMAP Stratified by HIV Group") +
    theme_minimal(base_size = 12) +
    scale_color_manual(values = anno_colours$HIV_Group)


 dev.off()

#Plot 4–UMAP Stratified by HPV Genotype

 tiff("UMAP_HPV_Group.tiff",
     width = tiff_width, height = tiff_height,
     res = tiff_res, compression = tiff_compression)


 ggplot(umap_df, aes(UMAP1, UMAP2, color = HPV_Group)) +
    geom_point(size = 3, alpha = 0.9) +
    labs(title = "UMAP Stratified by HPV Group") +
    theme_minimal(base_size = 12) +
    scale_color_manual(values = anno_colours$HPV_Group)


 dev.off()
```



8.2) Phenotypic Stratification of TME_Driver Model Methylome Clusters

To establish clinical relevance, molecularly derived clusters are evaluated against clinical and phenotypic annotations. This reveals whether molecular states correspond to patient characteristics, disease severity, immune contexture, or outcome. Overlaying clinical metadata on UMAP clusters shows clear enrichment of specific clinical groups within distinct clusters, implying that latent molecular states have clinical meaning. This confirms that stratification is not only computationally valid but clinically interpretable.



8.3) Functional Enrichment Analysis of TME_Driver Model Methylation Clusters

Following the identification of tumour subpopulations based on reconstructed joint TME and Driver gene programmes, functional enrichment analysis was performed to determine whether these clusters represent biologically coherent and mechanistically interpretable states. DNA methylation alterations encode transcriptional regulatory shifts, immune—stromal engagement, proliferative control, stress response, and pathway rewiring. Therefore, functional enrichment enables us to link methylation—defined clusters to biological processess and tumour ecosystem behaviours, determine whether clusters reflect immune—inflamed vs immune—cold vs stromal—driven etc., states, identify pathways that may underpin therapeutic vulnerabilities, and validate whether methylation signatures correspond to meaningful TME biology rather than atitrary clustering.

8.3.1) Data Preparation

1. **Deriving Cluster—Specific Differential Methylation Biomarkers**

For each Driver_TME cluster, CpGs were compared against all other tumours combined to compare cluster—specific methylation deviation. For each cluster C_k mean β-values were computed across all samples within the cluster, these were contrasted against mean of the remaining tumours, resulting in a ranked differential methylation vector per

cluster. This produced biologically meaningful directional biomarkers representing CpGs relatively hyper—or hypomethylated within each TME state.

```
driver_cor_long <- driver_cor_results_annotated %>%
   select(
       Patient,
       CellLine,
       Correlation
   ) %>%
   mutate(
       Patient    = as.character(Patient),
       CellLine   = as.character(CellLine),
       Correlation = as.numeric(Correlation)
   )


tme_cor_long <- tme_cor_results_annotated %>%
   select(
       Patient,
       CellLine,
       Correlation
   ) %>%
   mutate(
       Patient = as.character(Patient),
       CellLine = as.character(CellLine),
       Correlation = as.numeric(Correlation)
   )

driver_mat <- driver_cor_long %>%
   tidyr::pivot_wider(
       names_from  = CellLine,
       values_from = Correlation
   ) %>%
   column_to_rownames("Patient") %>%
   as.matrix()


# Remove patients or models with missing values
driver_mat <- driver_mat[
   rowSums(is.na(driver_mat)) == 0,
   colSums(is.na(driver_mat)) == 0
]


tme_mat <- tme_cor_long %>%
   pivot_wider(
       names_from  = CellLine,
       values_from = Correlation
   ) %>%
   column_to_rownames("Patient") %>%
   as.matrix()


# Remove cell lines or patients with missing values
tme_mat <- tme_mat[
   rowSums(is.na(tme_mat)) == 0,
   colSums(is.na(tme_mat)) == 0
]
common_patients <- intersect(rownames(tme_mat), rownames(driver_mat))
common_models   <- intersect(colnames(tme_mat), colnames(driver_mat))
```

```
 tme_driver_joint <- list(
    TME    = tme_mat[common_patients, common_models],
    Driver = driver_mat[common_patients, common_models]
 )


set.seed(123)


 # Joint similarity representation:
 # We average TME + Driver matrices to form an integrated similarity
 joint_mat <- (tme_driver_joint$TME + tme_driver_joint$Driver) / 2

 rph_joint <- Rphenograph(joint_mat, k = 30)
#Run Rphenograph starts:
#  -Input data of 100 rows and 15 columns
#  -k is set to 30
#  Finding nearest neighbors...DONE ~ 0.019 s
#  Compute jaccard coefficient between nearest-neighbor sets...DONE ~ 0.049 s
#  Build undirected graph from the weighted links...DONE ~ 0.048 s
#  Run louvain clustering on the graph ...DONE ~ 0.026 s
#Run Rphenograph DONE, totally takes 0.142000000000053s.
#  Return a community class
#  -Modularity value: 0.5876823
#  -Number of clusters: 3>
 # Inspect output
 #Run Rphenograph starts:
 #  -Input data of <Npatients> rows and <Nmodels> columns
 #  -k is set to 30
 #  Finding nearest neighbors...DONE
 #  Compute jaccard coefficient...DONE
 #  Build undirected graph...DONE
 #  Run Louvain clustering...DONE
 #Run Rphenograph DONE
 #  Return a community class


 # Extract cluster membership
 joint_clusters <- factor(membership(rph_joint[[2]]))


 # Retrieve modularity score
 joint_modularity_value <- rph_joint[[2]]$modularity
 print(joint_modularity_value)
#[1] 0.5876823


Implementation:

#Assume you have a clinical patient table: clinical_patient_aligned
patient_meta_joint <- data.frame(
    Patient = rownames(joint_mat),
    Cluster = joint_clusters
) %>%
    left_join(
        clinical_patient_aligned,
        by = c("Patient" = "Sentrix_ID")  # adjust column name if needed
    )
library(dplyr)
```

```
cluster_ranks <- list()
for (cl in unique(patient_meta_joint$Cluster)) {
    group_samples <- patient_meta_joint %>%
        filter(Cluster == cl) %>%
        pull(Patient)
    other_samples <- patient_meta_joint %>%
        filter(Cluster != cl) %>%
        pull(Patient)
    diff_vec <- rowMeans(beta_mat[, group_samples], na.rm = TRUE) -
        rowMeans(beta_mat[, other_samples], na.rm = TRUE)
    ranks <- sort(diff_vec, decreasing = TRUE)
    cluster_ranks[[paste0("C", cl)]] <- ranks
}
names(cluster_ranks)
#[1] "C1" "C2" "C3"
```

2. Gene-Level Mapping of Methylation Signals

Since GO/KEGG operate at gene level, CpGs were mapped to their annotated genes. CpGs were assigned to genes using Illumina annotation, multiple CpGs mapping to the same gene aggregated by mean methylation shift, resulting in a ranked gene list per cluster capturing gene—level regulatory methylation bias. This conversion ensures enrichment operates on biologically interpretable gene entities while preserving methylation—derived signal.

Implementation:

```
library(dplyr)
# Prepare annotation mapping CpG -> Gene
cpg_to_gene <- anno_df %>%
    tidyr::separate_rows(UCSC_RefGene_Name, sep = ";") %>%  # split multi-gene CpGs
    select(CpG = Name, Gene = UCSC_RefGene_Name) %>%
    distinct()
# Map CpGs to genes in cluster_ranks
cluster_ranks_genes <- list()
for (cl in names(cluster_ranks)) {
    ranks <- cluster_ranks[[cl]]  # named vector of CpG -> diff
    rank_df <- data.frame(
        CpG = names(ranks),
        Score = as.numeric(ranks),
        stringsAsFactors = FALSE
    )
    # Join with annotation to get gene names
    rank_df <- rank_df %>%
        left_join(cpg_to_gene, by = "CpG") %>%
        filter(!is.na(Gene))  # remove CpGs without gene annotation
    # Aggregate if multiple CpGs map to same gene: take mean score
    rank_gene_vec <- rank_df %>%
        group_by(Gene) %>%
        summarise(Score = mean(Score, na.rm = TRUE)) %>%
        arrange(desc(Score))
    cluster_ranks_genes[[cl]] <- setNames(rank_gene_vec$Score, rank_gene_vec$Gene)
}
# Inspect
names(cluster_ranks_genes)
```

```
> head(cluster_ranks_genes[[1]])
LOC644554  DEFB103A  DEFB103B LINC00698   MIR2681     OR4X2
0.1954976 0.1899270 0.1899270 0.1821703 0.1787614 0.1764558
>
> cluster_ranks_genes <- lapply(cluster_ranks_genes, function(v){
+
+    # remove NA names
+    v <- v[!is.na(names(v))]
+    # remove empty string names
+    v <- v[names(v) != ""]
+    # ensure numeric + sorted decreasing
+    v <- sort(v, decreasing = TRUE)
+    return(v)
+ })
>
> any(names(cluster_ranks_genes[[1]]) == "")
[1] FALSE
> #[1] FALSE
> any(is.na(names(cluster_ranks_genes[[1]])))
[1] FALSE
> #[1] FALSE

> library(dplyr)
> library(purrr)
>
> go_list <- go_df %>%
+    filter(!is.na(gene_symbol), gene_symbol != "") %>%
+    group_split(gs_name) %>%
+    set_names(map_chr(., ~ unique(.x$gs_name))) %>%
+    map(~ unique(.x$gene_symbol))
>
> stats <- cluster_ranks_genes[[cl]]
> stats <- clean_rank_names(stats)
>
> stats <- stats[!is.na(stats)]
> stats <- stats[!duplicated(names(stats))]
> stats <- sort(stats, decreasing = TRUE)
>
> library(fgsea)
>
> go_res <- list()
>
> for (cl in names(cluster_ranks_genes)) {
+
+    stats <- cluster_ranks_genes[[cl]]
+    stats <- clean_rank_names(stats)
+    stats <- stats[!is.na(stats)]
+    stats <- stats[!duplicated(names(stats))]
+    stats <- sort(stats, decreasing = TRUE)
+
+    go_res[[cl]] <- fgsea(
+        pathways = go_list,
+        stats    = stats,
+        minSize  = 15,
+        maxSize  = 500
+    ) %>% arrange(padj)
+ }
There were 12 warnings (use warnings() to see them)
```

```
> dim(go_res$C1)
[1] 4150    8
> head(go_res$C1)
                                                                  pathway
                                                                   <char>
1:                                 GOBP_DETECTION_OF_CHEMICAL_STIMULUS
2:            GOBP_DETECTION_OF_STIMULUS_INVOLVED_IN_SENSORY_PERCEPTION
3:                         GOBP_SENSORY_PERCEPTION_OF_CHEMICAL_STIMULUS
4:                                   GOBP_SENSORY_PERCEPTION_OF_SMELL
5: GOBP_HOMOPHILIC_CELL_ADHESION_VIA_PLASMA_MEMBRANE_ADHESION_MOLECULES
6:                                       GOBP_CELL_FATE_SPECIFICATION
           pval         padj  log2err         ES       NES  size  leadingEdge
          <num>        <num>    <num>      <num>     <num> <int>       <list>
1: 1.000000e-50 9.982500e-48       NA  0.7711034  2.307382   454 OR4X2, O....
2: 1.000000e-50 9.982500e-48       NA  0.7520222  2.254629   491 OR4X2, O....
3: 1.000000e-50 9.982500e-48       NA  0.7706675  2.307226   472 OR4X2, O....
4: 1.000000e-50 9.982500e-48       NA  0.7943211  2.374182   400 OR4X2, O....
5: 7.217092e-19 5.763569e-16 1.123915 -0.6370036 -2.540574   167 PCDH11Y,....
6: 1.113907e-17 7.413050e-15 1.086441 -0.7025040 -2.658686   110 ASCL1, S....
```

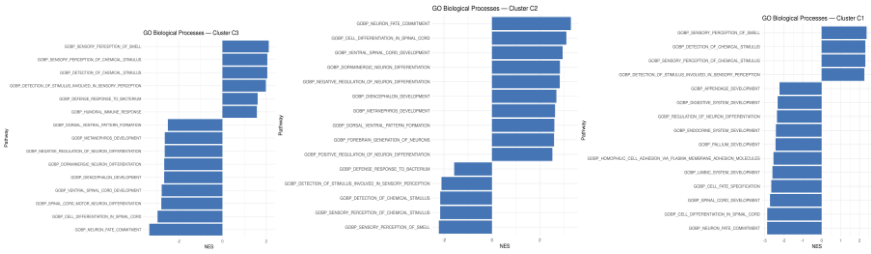
8.3.2) Gene Ontology (GO) Biological Process Enrichment

GO enrichment was peformed using FGSEA applied to the ranked methylation—derived gene scores. Utilises full ranked distributions rather than arbitrary thresholds, detect coordinated pogramme shifts, enhance sensitivity to subtle regulatory differences, and significance threshold (FDR—ajusted p < 0.05). For each cluster, the top enriched biological processes were extracted and visualised as high—resolution barplots.

Implementation:

```
library(fgsea)

go_res <- list()
  for (cl in names(cluster_ranks_genes)) {


    go_res[[cl]] <- fgsea(
        pathways = go_pathways,
        stats = cluster_ranks_genes[[cl]],
        minSize = 15,
        maxSize = 500
    ) %>% arrange(padj)
 }
#There were 12 warnings (use warnings() to see them)
 lapply(go_res, nrow)
#$C1
#[1] 4150
#$C2
#[1] 4150
#$C3
#[1] 4150
 library(ggplot2library(dplyr)


 # make output folder
 out_dir <- "GO_Biological_Processes_Results"
 dir.create(out_dir, showWarnings = FALSE)
```

```
for (cl in names(go_res)) {

   top_go <- go_res[[cl]] %>%
       filter(padj < 0.05) %>%
       arrange(padj) %>%
       slice_head(n = 15)

   # skip if no significant pathways
   if (nrow(top_go) == 0) next

   tiff(
       filename = file.path(out_dir, paste0("GO_Biological_Processes_", cl, ".tiff")),
       width = 4500,
       height = 4500,
       res = 600,
       compression = "lzw"
   )

   p <- ggplot(top_go, aes(x = reorder(pathway, NES), y = NES)) +
       geom_col(fill = "#4575B4") +
       coord_flip() +
       labs(
           title = paste("GO Biological Processes — Cluster", cl),
           x = "Pathway",
           y = "NES"
       ) +
       theme_minimal(base_size = 12)


   dev.off()
```



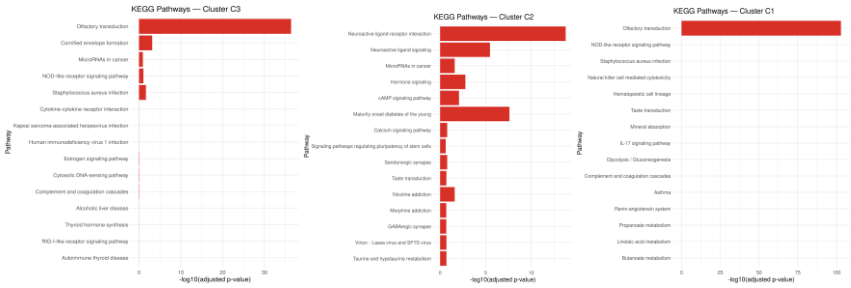
8.3.3) KEGG Pathway Enrichment

To investigate canonical cancer signaling and metabolic pathways, the top 500 genes per cluster were selected based on ranking, genes were mapped to ENTREZ IDs, pathway enrichment was performed using enrichKEGG, significance threshold (adjusted $p < 0.05$), and the top 15 KEGG pathways per cluster were visualised. KEGG analysis complements GO by revealing oncogenic signaling networks, immune signaling, stromal programmes, proliferation control, and metabolic rewiring, and stress—response systems enriched within each TME subtype.

Implementation:

```
library(clusterProfiler)
library(org.Hs.eg.db)
library(ggplot2)
library(dplyr)

# Output folder for plots
out_dir <- "KEGG_Cluster_Enrichment"
dir.create(out_dir, showWarnings = FALSE)

# Initialize result list
kegg_res <- list()

# Loop over clusters
for(cl in names(cluster_ranks_genes)) {

   # Take top 500 genes per cluster
   top_genes <- names(sort(cluster_ranks_genes[[cl]], decreasing = TRUE)[1:500])

   # Map to ENTREZ IDs
   entrez_genes <- bitr(top_genes,
                        fromType = "SYMBOL",
                        toType = "ENTREZID",
                        OrgDb = org.Hs.eg.db) %>%
       dplyr::distinct(ENTREZID) %>% pull(ENTREZID)

   cat(cl, "mapped genes:", length(entrez_genes), "\n")

   if(length(entrez_genes) == 0) next

   # KEGG enrichment
   kegg_enrich <- enrichKEGG(
       gene = entrez_genes,
       organism = "hsa",
       minGSSize = 10,
       pvalueCutoff = 0.05
   )

   if(is.null(kegg_enrich) || nrow(kegg_enrich) == 0) {
       cat("No significant pathways for", cl, "\n")
       next
   }

   # Store results
   kegg_res[[cl]] <- kegg_enrich

   # Select top 15 pathways by adjusted p-value
   top_kegg <- kegg_enrich@result %>%
       arrange(p.adjust) %>%
       slice_head(n = 15)

   # Convert GeneRatio for plotting
   top_kegg <- top_kegg %>%
       mutate(GeneRatio_num = sapply(strsplit(GeneRatio, "/"),
```

```
                                    function(x) as.numeric(x[1])/as.numeric(x[2])))

    # Plot
    plot_file <- file.path(out_dir, paste0("KEGG_", cl, ".tiff"))
    tiff(filename = plot_file, width = 4500, height = 4500, res = 600, compression = "lzw")

    p <- ggplot(top_kegg, aes(x = reorder(Description, GeneRatio_num), y = -log10(p.adjust))) +
        geom_col(fill = "#D73027") +
        coord_flip() +
        labs(title = paste("KEGG Pathways — Cluster", cl),
             x = "Pathway", y = "-log10(adjusted p-value)") +
        theme_minimal(base_size = 12)

    dev.off()
    cat("Plot saved to:", plot_file, "\n")
 }
#'select()' returned 1:1 mapping between keys and columns
#C1 mapped genes: 409
#Plot saved to: KEGG_Cluster_Enrichment/KEGG_C1.tiff
#'select()' returned 1:1 mapping between keys and columns
#C2 mapped genes: 414
#Plot saved to: KEGG_Cluster_Enrichment/KEGG_C2.tiff
#'select()' returned 1:1 mapping between keys and columns
#C3 mapped genes: 407
#Plot saved to: KEGG_Cluster_Enrichment/KEGG_C3.tiff
```



9. Functional Analysis of Methylome Intra-Cluster

Tutorial 9 extends the methylation-defined tumour microenvironment (TME) framework by systematically interrogating within-cluster epigenetic heterogeneity for each previously defined methylation endotype (*C1–C3*). While **Tutorial 8** focused on inter-cluster contrasts to identify signatures that distinguish tumour classes, this tutorial addresses a complementary and biologically critical question: whether additional, functionally meaningful sub-states exist within each cluster. By restricting differential methylation analysis to tumours belonging to a single cluster and performing case–control comparisons within that fixed epigenetic context, Tutorial 9 explicitly decouples secondary biological modifiers from the dominant tumour state captured by clustering. HIV status is used as a worked example to demonstrate how an exogenous modifier reshapes DNA methylation programmes within an otherwise homogeneous tumour microenvironment. This design enables isolation of cluster-specific HIV-associated

DMPs and DMRs, identification of intra-cluster immune, stromal, and epithelial sub-programmes, and discovery of pathway-level enrichments (GO, KEGG, Hallmark) driven by residual heterogeneity rather than cluster identity. Conceptually, **Tutorial 9** shifts the analysis from tumour classification toward subtype refinement, revealing latent sub-states (for example, hyper-inflamed versus baseline-inflamed tumours within C1) that may influence biological interpretation, therapeutic vulnerability, and experimental model alignment. More broadly, this tutorial provides a generalisable blueprint for assessing how clinical covariates, infections, co-pathogens, or environmental exposures modulate DNA methylation patterns and tumour microenvironment– and driver-associated programmes within defined tumour endotypes.

Relationship to Inter-Cluster Analysis

In Tutorial 8, orthogonal inter-cluster contrasts were used to derive methylation-associated gene signatures that define each cluster's core biological identity. These contrasts—C1 vs (C2+C3), C2 vs (C1+C3), and so forth—capture between-cluster variance and yield stable, reproducible TME_Driver archetypes. Tutorial 9 deliberately inverts this logic: instead of contrasting clusters against one another, it holds cluster identity constant and interrogates residual variance within each group. This ensures that detected DMPs and downstream functional enrichments reflect modifier-specific biology rather than re-discovery of the clustering signal itself. Together, Tutorials 8 and 9 provide a hierarchical epigenetic framework in which global tumour states are first defined by inter-cluster methylation structure and subsequently refined through intra-cluster functional stratification.

9.1) Intra-Cluster Differential Expression Analysis (DEA)

The following analytical steps implement a cluster-conditioned differential methylation strategy designed to isolate HIV-associated epigenetic effects within, rather than between, methylation-defined tumour endotypes. This design follows the logic of Tutorial 3 but introduces an additional stratification layer by restricting each model fit to a single cluster at a time. First, all unique tumour clusters are identified from the joint patient metadata and iterated over in a controlled loop. For each cluster, only samples assigned to that cluster are retained, ensuring that downstream contrasts are performed within a homogeneous tumour microenvironmental context. This prevents dominant inter-cluster methylation differences from confounding the analysis and allows detection of modifier-driven variation that would otherwise be masked. Within each cluster, the β-value matrix is subset to the corresponding patients, and HIV status is explicitly realigned to the methylation matrix columns. Converting HIV status to a clean, ordered factor enforces a consistent reference level (HIV-negative), enabling interpretable directionality of logFC estimates across clusters. A strict sanity check confirms perfect alignment between phenotype labels and methylation data, safeguarding against silent sample mismatches. Linear modelling is then performed using limma, with empirical Bayes moderation applied to stabilise variance estimates in the context of reduced within-cluster sample sizes. This is particularly important for smaller clusters, where conventional variance estimation would be unstable. Differentially methylated positions are extracted for the HIV-positive coefficient, and filtering by both statistical significance and effect size ($|\Delta\beta| \geq 0.15$, $P < 0.01$) ensures retention of biologically meaningful

methylation changes rather than nominally significant but negligible effects. The resulting DMP sets are stored separately for each cluster, enabling direct comparison of HIV-associated epigenetic burden across tumour endotypes.

Implementation:

```
library(limma)
library(dplyr)


 clusters <- sort(unique(patient_meta_joint$Cluster))
 dmp_results <- list()


 for (cl in clusters) {


    message("Processing cluster: ", cl)


    # -------------------------
    # Subset samples in cluster
    # -------------------------
    patients_cl <- patient_meta_joint$Patient[
        patient_meta_joint$Cluster == cl
    ]


    # subset beta matrix
    beta_cl <- beta_mat[, patients_cl]


    # -------------------------
    # HIV status aligned to columns of beta_cl
    # -------------------------
    hiv_status <- patient_meta_joint$HIV_status[
        match(colnames(beta_cl), patient_meta_joint$Patient)
    ]


    # clean factor
    hiv_status <- tolower(hiv_status)
    hiv_status <- factor(hiv_status, levels = c("negative", "positive"))

    # sanity check
    stopifnot(length(hiv_status) == ncol(beta_cl))


    # -------------------------
    # limma model
    # -------------------------
    design <- model.matrix(~ hiv_status)


    fit <- lmFit(beta_cl, design)
    fit <- eBayes(fit)


    dmps <- topTable(
        fit,
        coef   = "hiv_statuspositive",
        adjust = "fdr",
        number = Inf
    )
```

```
    dmps <- dmps %>%
        filter(abs(logFC) >= 0.15, P.Value < 0.01)

    dmp_results[[paste0("C", cl)]] <- dmps
 }
#Processing cluster: 1
#Processing cluster: 2
#Processing cluster: 3

 # check DMP counts
 lapply(dmp_results, dim)
#$C1
#[1] 7181    6

#$C2
#[1] 1825    6

#$C3
#[1] 44432     6
```

9.1.1 Visualisations

Subsequent visualisation steps translate these cluster-specific DMP tables into volcano plots, providing an intuitive summary of intra-cluster methylation dynamics. By classifying CpGs as hyper- or hypomethylated in HIV-positive tumours and plotting effect size against statistical support, these figures reveal both the magnitude and directionality of HIV-associated epigenetic reprogramming within each cluster. Labelling the most significant CpGs highlights candidate loci driving intra-cluster heterogeneity and facilitates downstream functional annotation.

```
 Implementation:
 library(ggplot2)
 library(dplyr)

 # Use C3 DMP table
 c3 <- dmp_results$C3

 # add status label
 c3 <- c3 %>%
    mutate(
        status = case_when(
            P.Value < 0.01 & logFC >= 0.15  ~ "Hypermethylated (HIV+)",
            P.Value < 0.01 & logFC <= -0.15 ~ "Hypomethylated (HIV+)",
            TRUE ~ "NS"
        ),
        neglogP = -log10(P.Value)
    )

 table(c1$status)

#Hypermethylated (HIV+)  Hypomethylated (HIV+)
```

```
#                    5826                    1355
```

```
# pick top significant hits to label
top_labels <- c3 %>%
   filter(status != "NS") %>%
   arrange(P.Value) %>%
   head(12)
```

logFC	AveExpr	t	P.Value	adj.P.Val	B	status	neglogP
-0.2171514	0.08090651	-8.459884	1.171373e-09	0.0002036486	11.99473	Hypomethylated (HIV+)	8.931305
-0.1586396	0.50782969	-8.387613	1.418346e-09	0.0002036486	11.80377	Hypomethylated (HIV+)	8.848218
-0.2803105	0.53471377	-8.318620	1.703602e-09	0.0002036486	11.62085	Hypomethylated (HIV+)	8.768632
-0.2138408	0.59957200	-8.312164	1.733121e-09	0.0002036486	11.60370	Hypomethylated (HIV+)	8.761171
-0.2795900	0.48866385	-8.267173	1.953820e-09	0.0002036486	11.48406	Hypomethylated (HIV+)	8.709115
-0.1607253	0.11381847	-8.238963	2.106592e-09	0.0002036486	11.40891	Hypomethylated (HIV+)	8.676420
0.2984996	0.80301659	8.215991	2.239941e-09	0.0002036486	11.34765	Hypermethylated (HIV+)	8.649763
0.2925453	0.83813036	8.136753	2.769523e-09	0.0002036486	11.13581	Hypermethylated (HIV+)	8.557595
-0.2051901	0.08227348	-8.125601	2.853667e-09	0.0002036486	11.10594	Hypomethylated (HIV+)	8.544597
-0.2497163	0.41255796	-8.053082	3.468137e-09	0.0002146885	10.91129	Hypomethylated (HIV+)	8.459904
-0.2009285	0.56548821	-8.048653	3.509759e-09	0.0002146885	10.89938	Hypomethylated (HIV+)	8.454723
-0.2302856	0.49491399	-7.968716	4.354891e-09	0.0002331504	10.68403	Hypomethylated (HIV+)	8.361023
-0.2114471	0.52056142	-7.900956	5.232005e-09	0.0002635595	10.50089	Hypomethylated (HIV+)	8.281332
-0.1577221	0.70017966	-7.870061	5.689621e-09	0.0002706888	10.41720	Hypomethylated (HIV+)	8.244917
0.2051151	0.77976101	7.840712	6.162005e-09	0.0002721401	10.33760	Hypermethylated (HIV+)	8.210278
-0.1896534	0.50923579	-7.829333	6.355697e-09	0.0002721401	10.30671	Hypomethylated (HIV+)	8.196837
-0.2698666	0.52721489	-7.801235	6.860949e-09	0.0002797849	10.23037	Hypomethylated (HIV+)	8.163616
0.2772147	0.83733441	7.764326	7.587249e-09	0.0002850007	10.12995	Hypermethylated (HIV+)	8.119916
0.2671098	0.85934738	7.746778	7.959512e-09	0.0002850007	10.08214	Hypermethylated (HIV+)	8.099114
0.2465531	0.70583361	7.745503	7.987259e-09	0.0002850007	10.07867	Hypermethylated (HIV+)	8.097602

```
# ----------------------
# Volcano Plot
# ----------------------
p_volcano <- ggplot(c3, aes(x = logFC, y = neglogP)) +
   geom_point(aes(color = status), size = 2, alpha = 0.9) +
   scale_color_manual(values = c(
       "Hypomethylated (HIV+)" = "#1B9E77",
       "Hypermethylated (HIV+)" = "#D95F02",
       "NS" = "grey70"
   )) +
   geom_vline(xintercept = c(-0.15, 0.15),
              linetype = "dashed", color = "black") +
   geom_hline(yintercept = -log10(0.05),
              linetype = "dashed", color = "black") +
   theme_minimal(base_size = 14) +
   labs(
       title = "Differential Methylation within Cluster C3 (HIV+ vs HIV-)",
       x = "Methylation logFC (HIV+ vs HIV-)",
       y = "-log10(P-value)",
       color = ""
   ) +
   theme(
       legend.position = "right",
       plot.title = element_text(face = "bold")
   ) +
   geom_text(
       data = top_labels,
       aes(label = rownames(top_labels)),
       size = 3,
       vjust = -0.7,
       check_overlap = TRUE
   )

p_volcano

> tiff(filename = "Volcano_C3_DMPs_Hiv.tiff",
+     width = 3500, height = 3500, res = 600, compression = "lzw")
> print(p_volcano)
```

```
> dev.off()
RStudioGD
        2
```

```
 library(ggplot2)
 library(dplyr)


 # Use C3 DMP table
 c2 <- dmp_results$C2


 # add status label
 c2 <- c2 %>%
    mutate(
        status = case_when(
            P.Value < 0.01 & logFC >= 0.15  ~ "Hypermethylated (HIV+)",
            P.Value < 0.01 & logFC <= -0.15 ~ "Hypomethylated (HIV+)",
            TRUE ~ "NS"
        ),
        neglogP = -log10(P.Value)
    )


 table(c2$status)


#Hypermethylated (HIV+)  Hypomethylated (HIV+)
#                   953                    872


 # pick top significant hits to label
 top_labels <- c2 %>%
    filter(status != "NS") %>%
    arrange(P.Value) %>%
    head(12)


 # ----------------------
 # Volcano Plot
 # ----------------------
 p_volcano <- ggplot(c2, aes(x = logFC, y = neglogP)) +
    geom_point(aes(color = status), size = 2, alpha = 0.9) +
    scale_color_manual(values = c(
        "Hypomethylated (HIV+)" = "#1B9E77",
        "Hypermethylated (HIV+)" = "#D95F02",
        "NS" = "grey70"
    )) +
    geom_vline(xintercept = c(-0.15, 0.15),
               linetype = "dashed", color = "black") +
    geom_hline(yintercept = -log10(0.05),
               linetype = "dashed", color = "black") +
    theme_minimal(base_size = 14) +
    labs(
        title = "Differential Methylation within Cluster C2 (HIV+ vs HIV-)",
        x = "Methylation logFC (HIV+ vs HIV-)",
        y = "-log10(P-value)",
        color = ""
    ) +
    theme(
        legend.position = "right",
        plot.title = element_text(face = "bold")
```

```
    ) +
    geom_text(
        data = top_labels,
        aes(label = rownames(top_labels)),
        size = 3,
        vjust = -0.7,
        check_overlap = TRUE
    )

 p_volcano
```

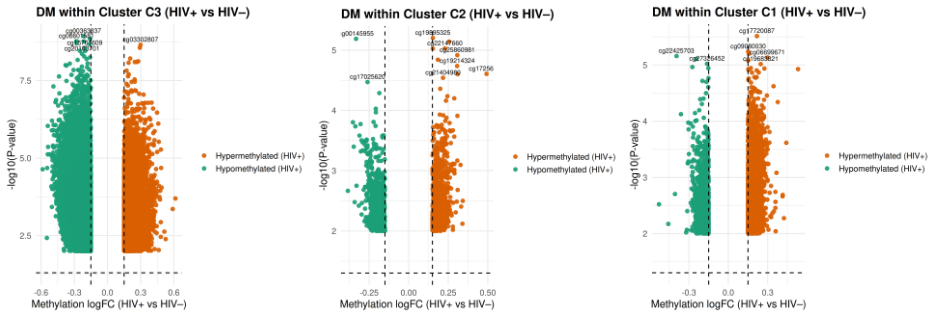
```
> tiff(filename = "Volcano_C2_DMPs_Hiv.tiff",
+     width = 3500, height = 3500, res = 600, compression = "lzw")
> print(p_volcano)
> dev.off()
RStudioGD
        2
#RStudioGD
#        2
```

```
 library(ggplot2)
 library(dplyr)


 # Use C1 DMP table
 c1 <- dmp_results$C1


 # add status label
 c1 <- c1 %>%
    mutate(
        status = case_when(
            P.Value < 0.01 & logFC >= 0.15  ~ "Hypermethylated (HIV+)",
            P.Value < 0.01 & logFC <= -0.15 ~ "Hypomethylated (HIV+)",
            TRUE ~ "NS"
        ),
        neglogP = -log10(P.Value)
    )


 table(c1$status)
```

```
#Hypermethylated (HIV+)  Hypomethylated (HIV+)
#                  5826                   1355
```

```
 # pick top significant hits to label
 top_labels <- c1 %>%
    filter(status != "NS") %>%
    arrange(P.Value) %>%
    head(12)


 # ----------------------
 # Volcano Plot
 # ----------------------
 p_volcano <- ggplot(c1, aes(x = logFC, y = neglogP)) +
    geom_point(aes(color = status), size = 2, alpha = 0.9) +
    scale_color_manual(values = c(
        "Hypomethylated (HIV+)" = "#1B9E77",
        "Hypermethylated (HIV+)" = "#D95F02",
```

```
        "NS" = "grey70"
    )) +
    geom_vline(xintercept = c(-0.15, 0.15),
               linetype = "dashed", color = "black") +
    geom_hline(yintercept = -log10(0.05),
               linetype = "dashed", color = "black") +
    theme_minimal(base_size = 14) +
    labs(
        title = "Differential Methylation within Cluster C1 (HIV+ vs HIV-)",
        x = "Methylation logFC (HIV+ vs HIV-)",
        y = "-log10(P-value)",
        color = ""
    ) +
    theme(
        legend.position = "right",
        plot.title = element_text(face = "bold")
    ) +
    geom_text(
        data = top_labels,
        aes(label = rownames(top_labels)),
        size = 3,
        vjust = -0.7,
        check_overlap = TRUE
    )

 p_volcano

  tiff(filename = "Volcano_C1_DMPs_Hiv.tiff",
     width = 3500, height = 3500, res = 600, compression = "lzw")
 print(p_volcano)
 dev.off()
#RStudioGD
#        2
```



Intra-cluster differential methylation analysis demonstrated that HIV infection induces distinct, endotype-specific epigenetic reprogramming within each methylation-defined tumour cluster rather than a uniform effect across cervical cancers. Within Cluster C3, the volcano plot revealed a large and asymmetric burden of HIV-associated DMPs, with a pronounced enrichment of hypomethylated CpGs in HIV-positive tumours alongside a substantial hypermethylated component. The wide logFC range (approximately −0.6 to +0.6) and high statistical significance (−log10 P-values exceeding 8 for top loci) indicate strong and coordinated HIV-driven epigenetic disruption within this endotype. The density and magnitude of signals suggest that C3 represents a highly HIV-responsive

tumour state, potentially corresponding to an immune-active or microenvironmentally plastic context in which viral co-infection strongly reshapes regulatory methylation landscapes. In contrast, Cluster C2 exhibited a markedly more constrained HIV-associated methylation profile. Both the number of significant DMPs and their effect sizes were reduced relative to C3, with logFC values largely confined within ±0.3 and lower overall −log10 P-values. Nevertheless, clear bidirectional methylation changes were still evident, indicating that HIV status remains biologically relevant even within this more epigenetically stable endotype. The balanced presence of hyper- and hypomethylated CpGs suggests targeted modulation of specific regulatory loci rather than broad epigenomic remodeling, consistent with a tumour state that is less permissive to widespread HIV-associated epigenetic shifts. Cluster C1 displayed an intermediate phenotype, with a robust number of HIV-associated DMPs and moderate-to-large effect sizes. The volcano plot showed strong enrichment of hypermethylated CpGs in HIV-positive tumours, accompanied by a distinct but smaller hypomethylated component. Several highly significant loci exceeded −log10 P-values of 5, indicating reproducible and biologically meaningful methylation changes within this cluster. Compared with C3, the narrower logFC distribution suggests a more constrained response, yet the signal remains substantially stronger than that observed in C2, positioning C1 as a partially HIV-responsive tumour state.

9..2 Intra-(DEA)-Cluster Tumour—Model Mapping

Having established that HIV induces strong yet endotype-specific differential methylation within clusters, this section aims to determine whether intra-cluster, HIV-associated epigenetic variation translates into meaningful differences in tumour–model concordance. Rather than mapping tumours to experimental models using genome-wide or pan-cluster feature sets, the analysis restricts the feature space to cluster-specific HIV-associated DMPs. This conditions tumour–model similarity on a fixed tumour endotype while allowing only HIV-driven variation to inform the mapping. This design explicitly controls for baseline tumour state and isolates how an exogenous modifier—here, HIV infection—reshapes epigenetic programmes within a given cluster and alters alignment with experimental systems. By applying the tumour–model concordance framework to cluster-restricted DMP sets, we test whether model suitability is stable within an endotype or whether HIV status induces systematic shifts in concordance, thereby refining experimental model prioritisation at a sub-endotype resolution. Collectively, the results demonstrate that intra-cluster, DEA-restricted mapping substantially sharpens tumour–model relationships, revealing that model suitability is not uniform even within a single methylation-defined endotype. HIV-associated epigenetic reprogramming within C3 materially alters tumour–model concordance, identifying experimental systems that preferentially represent HIV-modified tumour states versus baseline C3 biology. This validates intra-cluster stratification as a critical extension beyond between-cluster comparisons, enabling context-aware experimental model selection that accounts simultaneously for tumour identity and clinically relevant modifiers such as HIV infection.

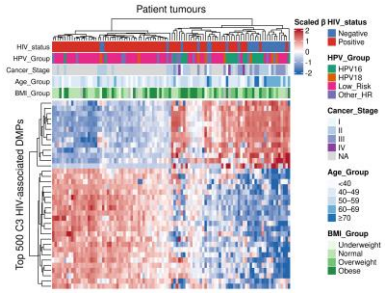
```
dmp_ids <- rownames(dmp_results$C3)

common_dmp_cpgs <- intersect(
```

```
    dmp_ids,
    intersect(rownames(beta_cell), rownames(beta_patient))
 )

 cat("DEA-restricted CpGs:", length(common_dmp_cpgs), "\n")
#DEA-restricted CpGs: 2310
 #DEA-restricted CpGs: 95378 (tutorial 4)

 beta_cell_dmp    <- beta_cell[common_dmp_cpgs, , drop = FALSE]
 beta_patient_dmp <- beta_patient[common_dmp_cpgs, , drop = FALSE]

 beta_cell_dmp    <- beta_cell[common_dmp_cpgs, , drop = FALSE]
 beta_patient_dmp <- beta_patient[common_dmp_cpgs, , drop = FALSE]


 cor_results_dmp <- data.frame()

 for (cell in colnames(beta_cell_dmp)) {
    for (pat in colnames(beta_patient_dmp)) {

        cor_val <- suppressWarnings(
            cor(
                beta_cell_dmp[, cell],
                beta_patient_dmp[, pat],
                method = "spearman",
                use = "pairwise.complete.obs"
            )
        )

        cor_results_dmp <- rbind(
            cor_results_dmp,
            data.frame(
                CellLine    = cell,
                Patient     = pat,
                Correlation = cor_val
            )
        )
    }
 }

 # Annotate with patient metadata
 cor_results_dmp_annotated <- cor_results_dmp %>%
    left_join(
        clinical_patient_aligned %>%
            select(Sentrix_ID, HIV_status, HPV_Genotype, Age, BMI, Cancer_Stage),
        by = c("Patient" = "Sentrix_ID")
    ) %>%
    left_join(
        cell_meta_aligned %>% select(Beta_Column, Cell_Symbol),
        by = c("CellLine" = "Beta_Column")
    ) %>%
    mutate(
        CellLine = ifelse(is.na(Cell_Symbol), CellLine, Cell_Symbol)
    ) %>%
```

```
   select(-Cell_Symbol)

head(cor_results_dmp_annotated)
```

CellLine	Patient	Correlation	HIV_status	HPV_Genotype	Age	BMI	Cancer_Stage
BOKU	203219670039_R07C01	-0.04966814	Negative	NA	60	23.875114784206	III
BOKU	203219640064_R04C01	-0.03234496	Negative	18	66	27	III
BOKU	203219640023_R03C01	0.21150081	Negative	59	76	20.690494543389	I
BOKU	203219730145_R08C01	0.28289453	Negative	18	55	25	III
BOKU	203220070185_R03C01	0.27532927	Negative	NA	68	30	I
BOKU	203219640064_R02C01	0.37906326	Negative	18	66	27	III
BOKU	203219750070_R08C01	-0.04961052	Negative	NA	60	23	III
BOKU	203219730045_R08C01	0.46068090	Negative	16	72	30.46875	I
BOKU	205841320033_R07C01	0.11607384	Negative	35	64	28	NA
BOKU	205841320110_R04C01	0.42548057	Negative	52	65	24.458063417102	II
BOKU	203219730045_R01C01	0.37261616	Negative	45	58	19.286155699467	III
BOKU	203219720180_R04C01	0.38520795	Negative	16	56	45	I
BOKU	203219730045_R04C01	0.30051262	Negative	18	50	17	II
BOKU	203219720078_R05C01	0.01140002	Negative	45	66	26	II
BOKU	203219640056_R04C01	0.10957606	Negative	35	65	24	II
BOKU	203219720220_R08C01	0.27706245	Negative	45	53	25	II
BOKU	203219720077_R06C01	0.18547551	Negative	16	86	22	III
BOKU	203219750121_R07C01	0.14031008	Negative	NA	70	25	II
BOKU	203219730145_R07C01	0.17942970	Negative	18	55	25	III
BOKU	203219750006_R02C01	-0.14642098	Negative	NA	48	20.077334919691	III

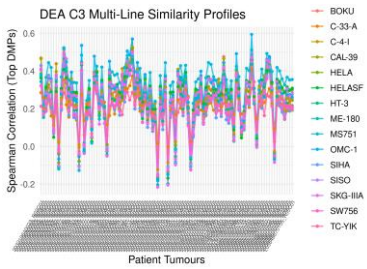
```
tiff(
   "Heatmap_Top500_C3_Clinical_Annotations.tiff",
   width = 4200,
   height = 3200,
   res = 600,
   compression = "lzw"
)

Heatmap(
   mat,
   name = "Scaled β",
   top_annotation = ha,
   show_row_names = FALSE,
   show_column_names = FALSE,
   cluster_rows = TRUE,
   cluster_columns = TRUE,
   row_title = "Top 500 C3 HIV-associated DMPs",
   column_title = "Patient tumours",
   col = colorRamp2(c(-2, 0, 2),
                    c("#2166AC", "white", "#B2182B"))
)



dev.off()
#RStudioGD
#         2
```



Restricting tumour–model similarity analysis to C3-specific HIV-associated DMPs yielded a compact yet highly informative feature space comprising 2,310 CpGs shared across patient tumours and experimental models, in sharp contrast to the much larger global DEA feature set used in earlier analyses. Heatmap visualisation of the top 500

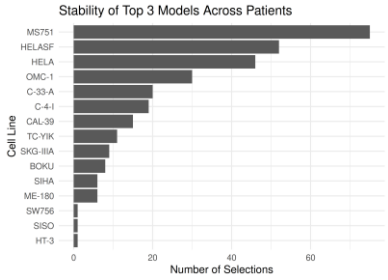
C3 HIV-associated DMPs revealed highly structured and coordinated methylation blocks across patient tumours, with clear segregation driven by HIV status despite all samples belonging to the same methylation-defined endotype. This demonstrates that substantial epigenetic heterogeneity persists within C3 and that HIV infection imposes a coherent secondary layer of regulation superimposed on the core cluster identity. Importantly, these methylation patterns aligned with additional clinical annotations—including HPV genotype, cancer stage, age, and BMI—indicating that intra-cluster HIV-associated methylation captures biologically meaningful tumour sub-contexts rather than residual noise.

```
 cor_results_dmp_annotated <- cor_results_dmp_annotated %>%
    mutate(

        Patient = as.character(Patient),
        CellLine = as.character(CellLine),
        Age = as.numeric(Age),
        BMI = as.numeric(BMI),
        Correlation = as.numeric(Correlation)
    )

 library(ggplot2)

 tiff(
    "DEA_C3_MultiLine_Similarity.tiff",
    width = 3500,
    height = 2500,
    res = 600,
    compression = "lzw"
 )

 ggplot(cor_results_dmp_annotated, aes(x = Patient, y = Correlation, group = CellLine, color =
CellLine)) +
    geom_line(alpha = 0.8, linewidth = 0.7) +
    geom_point(size = 1) +
    labs(
        title = "DEA C3 Multi-Line Similarity Profiles",
        x = "Patient Tumours",
        y = "Spearman Correlation (Top DMPs)"
    ) +
    theme_minimal(base_size = 12) +
    theme(
        axis.text.x = element_text(angle = 60, hjust = 1, size = 6),
        legend.position = "right"
    )

 dev.off()
#RStudioGD
#        2
```



DEA—Restricted Spearman correlation profiling between C3 tumours and cervical cancer experimental models further revealed non—random and structured tumour—model concordance patterns. The multi-line similarity plot showed that individual models exhibited recurrent peaks and troughs across patients, rather than flat or overlapping profiles, indicating selective alignment with subsets of tumours within the same cluster. Several models consistently achieved higher correlations across large fractions of patients, while others displayed pronounced patient—specific variability or uniformly low concordance, supporting the existence of HIV—modulated sub—states within C3 that are differentially represented in vitro. Notably, correlation values spanned both weak and moderately strong positive ranges, highlighting that restricting to HIV—associated CpGs amplifies biologically relevant signal while suppressing background similarity driven by shared endotype structure.

9.3 Model Performance



94 Functional Enrichment

Next pathway enrichment allows us to interpret pathway and epigenetic regulation in HIV+—vs—HIV- in C1, as well as mitochondrial, metabolic or exhaustion programs unique to the HIV influence;

```
> library(dplyr)
> library(ggplot2)
> library(forcats)
>
> # choose significance + number of pathways to show
> sig_go_C3 <- go_res_C3 %>%
+    filter(padj < 0.05) %>%
+    arrange(desc(NES))
>
> # choose top N positively enriched + optionally negatives
> topN <- 20
> plot_go_C3 <- sig_go_C3 %>%
```

```
+    slice_head(n = topN) %>%
+    mutate(
+        pathway = gsub("^GOBP_", "", pathway),
+        pathway = gsub("_", " ", pathway),
+        pathway = stringr::str_to_title(pathway),
+        pathway = fct_reorder(pathway, NES)
+    )
>
> p_go_bar_C3 <- ggplot(plot_go_C3,
+                       aes(x = NES, y = pathway, fill = padj)) +
+    geom_col() +
+    scale_fill_viridis_c(direction = -1, option = "C") +
+    labs(
+        title = "GO Biological Processes Enriched in C3",
+        x = "Normalized Enrichment Score (NES)",
+        y = "GO Biological Process",
+        fill = "FDR (padj)"
+    ) +
+    theme_minimal(base_size = 12) +
+    theme(
+        panel.grid.minor = element_blank(),
+        axis.text.y = element_text(size = 10),
+        plot.title = element_text(face = "bold", size = 14)
+    )
>
> p_go_bar_C3
>
> tiff(
+    "GO_FGSEA_C3_BarPlot.tiff",
+    width = 3800,
+    height = 3000,
+    res = 600,
+    compression = "lzw"
+ )
>
> print(p_go_bar_C3)
> dev.off()
RStudioGD
        2
```

9.5) Endotype-Specific Functional Reprogramming and HIV-Stratified Tumour Structure

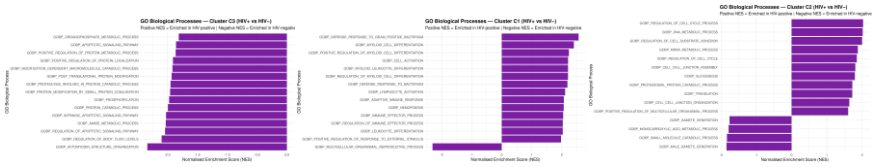
Following endotype-resolved differential methylation and tumour–model mapping, functional enrichment analysis was performed to contextualise HIV-associated epigenetic alterations within biological pathways. By analysing Gene Ontology (GO) Biological Process enrichment separately within each methylation-defined tumour cluster, this step assesses whether HIV-driven methylation changes converge on shared or distinct functional programmes across tumour endotypes. In parallel, joint integration of TME- and driver-restricted features was projected into low-dimensional space to evaluate whether combined contextual and oncogenic signals stratify tumours according to HIV status. Together, these analyses test the central premise that viral

status interacts with tumour endotype to generate structured and biologically coherent epigenetic reprogramming, rather than uniform global effects across all tumours.

The following data is required for this section of the tutorial;

```
> library(dplyr)
>
> top_go_C3 <- go_res_C3_HIV %>%
+    filter(padj < 0.05) %>%        # FDR 5%
+    arrange(padj) %>%
+    slice_head(n = 10) %>%         # adjust if you want 15
+    select(pathway, NES, padj, leadingEdge)
>
> top_go_C3
                                                          pathway       NES
                                                            <char>     <num>
 1:                      GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.336517
 2:                                GOBP_AMIDE_METABOLIC_PROCESS -2.027828
 3:                             GOBP_APOPTOTIC_SIGNALING_PATHWAY -1.856528
 4:                   GOBP_INTRINSIC_APOPTOTIC_SIGNALING_PATHWAY -2.023209
 5: GOBP_MODIFICATION_DEPENDENT_MACROMOLECULE_CATABOLIC_PROCESS -1.915996
 6:                      GOBP_ORGANOPHOSPHATE_METABOLIC_PROCESS -1.810090
 7:                                      GOBP_PHOSPHORYLATION -1.971936
 8:            GOBP_POSITIVE_REGULATION_OF_PROTEIN_LOCALIZATION -1.908064
 9:       GOBP_POSITIVE_REGULATION_OF_PROTEIN_METABOLIC_PROCESS -1.862347
10:                GOBP_POST_TRANSLATIONAL_PROTEIN_MODIFICATION -1.929916
          padj  leadingEdge
         <num>       <list>
 1: 0.02295496 ARHGEF15....
 2: 0.02295496 GDA, DGA....
 3: 0.02295496 TP53BP1,....
 4: 0.02295496 TP53BP1,....
 5: 0.02295496 LAPTM5, ....
 6: 0.02295496 GDA, DGA....
 7: 0.02295496 TAB2, PI....
 8: 0.02295496 FNTA, CH....
 9: 0.02295496 EEF2, TM....
10: 0.02295496 LAPTM5, ....

> library(purrr)
> library(tidyr)
>
> go_leading_genes_C3 <- top_go_C3 %>%
+    mutate(leadingEdge = lapply(leadingEdge, as.character)) %>%
+    unnest(leadingEdge) %>%
+    rename(Gene = leadingEdge)
>
> head(go_leading_genes_C3)
# A tibble: 6 × 4
  pathway                                NES   padj Gene
  <chr>                                <dbl>  <dbl> <chr>
1 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ARHGEF15
2 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ADPRHL1
3 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 OBSCN
4 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 TACR1
5 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 OBSL1
6 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 MKKS
```

```
> go_gene_cpg_map_C3 <- go_leading_genes_C3 %>%
+    left_join(gene_to_cpg, by = "Gene") %>%
+    filter(!is.na(CpG))
>

> go_gene_cpg_map_C3 %>% head()
# A tibble: 6 × 5
  pathway                                NES   padj Gene     CpG
  <chr>                                <dbl>  <dbl> <chr>    <list>
1 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ARHGEF15 <chr [28]>
2 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ADPRHL1  <chr [71]>
3 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 OBSCN    <chr [166]>
4 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 TACR1    <chr [45]>
5 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 OBSL1    <chr [55]>
6 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 MKKS     <chr [31]>


> go_cpg_sets_C3 <- go_gene_cpg_map_C3 %>%
+    unnest(CpG) %>%
+    distinct()
>
> head(go_cpg_sets_C3)
# A tibble: 6 × 5
  pathway                                NES   padj Gene     CpG
  <chr>                                <dbl>  <dbl> <chr>    <chr>
1 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ARHGEF15 cg17212698
2 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ARHGEF15 cg01869428
3 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ARHGEF15 cg17658213
4 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ARHGEF15 cg20038213
5 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ARHGEF15 cg23843103
6 GOBP_ACTOMYOSIN_STRUCTURE_ORGANIZATION -2.34 0.0230 ARHGEF15 cg19537511

> MAG_C3_HIV <- go_cpg_sets_C3 %>%
+    group_by(pathway) %>%
+    summarise(CpGs = list(unique(CpG)))
>
> mag_cpgs_C3 <- MAG_C3_HIV %>%
+    pull(CpGs) %>%
+    unlist() %>%
+    unique()
>
> length(mag_cpgs_C3)
[1] 53508

> common_mag_cpgs <- intersect(
+    mag_cpgs_C3,
+    intersect(rownames(beta_cell), rownames(beta_patient))
+ )
>
> cat("MAG-restricted CpGs:", length(common_mag_cpgs), "\n")
MAG-restricted CpGs: 27018
```



9.61 Feature Harmonisation and Metadata Normalisation

The construction of tme_df, driver_df, and dmp_df standardises correlation outputs across tutorials into a unified schema. Explicit type coercion ensures numerical stability and prevents downstream modelling artefacts, enabling direct comparison across feature spaces.

```
to_numeric_safe <- function(x){
suppressWarnings(as.numeric(x))

}
tme_df <- tme_cor_results_annotated %>%
mutate(

   Feature_Set = "TME",

   Age = to_numeric_safe(Age),

   BMI = to_numeric_safe(BMI),

   Cancer_Stage = as.character(Cancer_Stage),

   HIV_status   = as.character(HIV_status),

   HPV_Genotype = as.character(HPV_Genotype)

)
driver_df <- driver_cor_results_annotated %>%
mutate(

   Feature_Set = "Driver",

   Age = to_numeric_safe(Age),

   BMI = to_numeric_safe(BMI),

   Cancer_Stage = as.character(Cancer_Stage),

   HIV_status   = as.character(HIV_status),

   HPV_Genotype = as.character(HPV_Genotype)
```

```
)

dmp_df <- cor_results_dmp_annotated %>%

mutate(

   Feature_Set = "DMP",

   Age = to_numeric_safe(Age),

   BMI = to_numeric_safe(BMI),

   Cancer_Stage = as.character(Cancer_Stage),

   HIV_status   = as.character(HIV_status),

   HPV_Genotype = as.character(HPV_Genotype)

)
```

9.6.2 Feature-Space Integration

bind_rows() merges the three restricted similarity spaces into a single long-format table, preserving feature identity through the Feature_Set variable. This step formalises the principle that distinct biological axes must be analysed jointly while remaining conceptually and biologically distinct, rather than being conflated into a single undifferentiated feature space.

```
combined_cor <- bind_rows(tme_df, driver_df, dmp_df)

combined_cor <- combined_cor %>%

arrange(Patient)
```

```
> glimpse(combined_cor)
#Rows: 4,500
#Columns: 9
#$ CellLine     <chr> "BOKU", "C-33-A", "C-4-I", "CAL-39", "HELA", "HELASF", "…
#$ Patient      <chr> "203219640023_R02C01", "203219640023_R02C01", "203219640…
#$ Correlation  <dbl> 0.6233134, 0.5013186, 0.6599855, 0.7142319, 0.6988170, 0…
#$ HIV_status   <chr> "Positive", "Positive", "Positive", "Positive", "Positiv…
#$ HPV_Genotype <chr> "16", "16", "16", "16", "16", "16", "16", "16", "16", "1…
#$ Age          <dbl> 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, 42, …
#$ BMI          <dbl> 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, …
#$ Cancer_Stage <chr> "II", "II", "II", "II", "II", "II", "II", "II", "II", "I…
#$ Feature_Set  <chr> "TME", "TME", "TME", "TME", "TME", "TME", "TME", "TME", …
> summary(combined_cor$Age)
#   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#  22.00   42.00   47.00   49.34   57.00   86.00
> table(combined_cor$Feature_Set)
```

```
#   DMP Driver    TME
#  1500   1500   1500
```

9.6.3 Consensus Label Construction

The consensus labelling logic resolves discordant model assignments by prioritising agreement across feature spaces, with biologically informed fallback rules. This avoids arbitrary averaging and reflects an emphasis on context-aware decision making rather than purely statistical aggregation.

```
train_df <- feature_mat %>%
        left_join(best_TME,    by = "Patient") %>%
        left_join(best_DMP,    by = "Patient") %>%
        left_join(best_Driver, by = "Patient")

train_df <- train_df %>%
mutate(
   Consensus_Label = dplyr::case_when(
       TME_Label == DMP_Label                   ~ TME_Label,
       TME_Label == Driver_Label                ~ TME_Label,
       DMP_Label == Driver_Label                ~ DMP_Label,
       TRUE ~ Driver_Label   # fallback priority
   )
)
```

9.6.4 Supervised Learning via Random Forest

The random forest model is selected for its robustness to correlated predictors and non-linear interactions. Repeated cross-validation quantifies stability, while minimal tuning prevents over-parameterisation, ensuring that performance reflects biological signal rather than model complexity.

```
library(caret)
set.seed(123)
```

```
model_data <- train_df %>%

select(TME, Driver, DMP, Consensus_Label) %>%


na.omit()

ctrl <- trainControl(method="repeatedcv", number=10, repeats=5)

rf_model <- train(

Consensus_Label ~ TME + Driver + DMP,


data = model_data,


method = "rf",


trControl = ctrl


)

#note: only 2 unique complexity parameters in default grid. Truncating the grid to 2 .
```

9.6.5 Model Evaluation and Interpretability

Confusion matrices, variable importance, class distributions, and prediction probability outputs collectively assess accuracy, robustness, feature contribution, and confidence. This multi-angle evaluation reflects the design philosophy of the framework: transparent and biologically interpretable model selection rather than black-box optimisation.

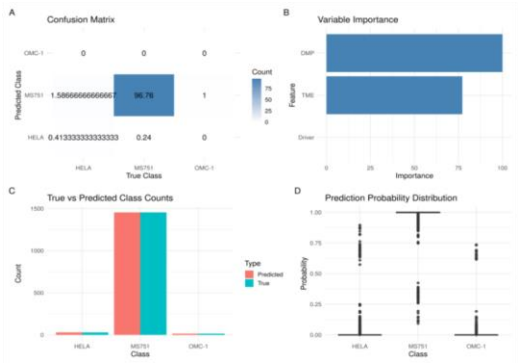
```
rf_model #Random Forest

#1500 samples 3 predictor 3 classes: 'HELA', 'MS751', 'OMC-1'
#No pre-processing Resampling: Cross-Validated (10 fold, repeated 5 times)
#Summary of sample sizes: 1351, 1349, 1350, 1350, 1350, 1350, ...
#Resampling results across tuning parameters:
#mtry Accuracy Kappa
# 2 0.9717398 0.2015135 3 0.9702687 0.2122813
#Accuracy was used to select the optimal model using the largest value. The final value used for
#the model was mtry = 2.
confusionMatrix(rf_model)

#Cross-Validated (10 fold, repeated 5 times) Confusion Matrix
#(entries are percentual average cell counts across resamples)
#     Reference

#Prediction HELA MS751 OMC-1 HELA 0.4 0.2 0.0 MS751 1.6 96.8 1.0 OMC-1 0.0 0.0 0.0
#Accuracy (average) : 0.9717

varImp(rf_model) #rf variable importance


#  Overall

#DMP 100.00 TME 77.23 Driver 0.00

preds <- predict(rf_model, model_data) table(preds, model_data$Consensus_Label)


#preds HELA MS751 OMC-1 HELA 30 0 0 MS751 0 1455 0 OMC-1 0 0 15
```

```
probs <- predict(rf_model, model_data, type="prob")

head(probs)

#HELA MS751 OMC-1 1 0.000 1.000 0.000 2 0.016 0.984 0.000 3 0.000 1.000 0.000 4 0.000 0.982 0.018
#5 0.000 1.000 0.000 6 0.000 0.890 0.110

results <- cbind(

train_df %>% select(Patient) %>% distinct(),


preds,


probs


)

#head(results) Patient preds HELA MS751 OMC-1 1 203219640023_R02C01 MS751 0.000 1.000 0.000 2
#203219640023_R03C01 MS751 0.016 0.984 0.000 3 203219640023_R04C01 MS751 0.000 1.000 0.000 4
#203219640023_R08C01 MS751 0.000 0.982 0.018 5 203219640056_R01C01 MS751 0.000 1.000 0.000 6
#203219640056_R04C01 MS751 0.000 0.890 0.110

table(model_data$Consensus_Label)

#HELA MS751 OMC-1
#   30  1455    15

table(preds)

preds

#HELA MS751 OMC-1
#   30  1455    15

library(ggplot2)
library(dplyr)
library(tidyr)
library(caret)
library(reshape2)
library(patchwork

##########################################
# ---------- PLOT A: Confusion Matrix -----
##########################################

cm <- confusionMatrix(rf_model)$table

cm_df <- as.data.frame(cm)

colnames(cm_df) <- c("Prediction","Reference","Freq")

plotA <- ggplot(cm_df, aes(x = Reference, y = Prediction, fill = Freq)) +

  geom_tile(color = "white") +
```

```
  geom_text(aes(label = Freq), size = 5) +
  scale_fill_gradient(low = "white", high = "steelblue") +
  theme_minimal(base_size = 14) +
  labs(title = "Confusion Matrix",
       x = "True Class",
       y = "Predicted Class",
       fill = "Count")


##########################################
# ----- PLOT B: Variable Importance -------
##########################################
var_df <- varImp(rf_model)$importance
var_df$Feature <- rownames(var_df)


plotB <- ggplot(var_df, aes(x = reorder(Feature, Overall), y = Overall)) +
  geom_col(fill = "steelblue") +
  coord_flip() +
  theme_minimal(base_size = 14) +
  labs(title = "Variable Importance",
       x = "Feature",
       y = "Importance")


##########################################
# ---- PLOT C: Class Distribution ---------
##########################################
true_counts <- as.data.frame(table(model_data$Consensus_Label))
pred_counts <- as.data.frame(table(preds))


colnames(true_counts) <- c("Class","True")
```

```
colnames(pred_counts) <- c("Class","Predicted")


dist_df <- merge(true_counts, pred_counts, by="Class")


dist_long <- dist_df %>%
  pivot_longer(cols = c(True, Predicted),
               names_to = "Type",
               values_to = "Count")


plotC <- ggplot(dist_long, aes(x = Class, y = Count, fill = Type)) +
  geom_col(position = "dodge") +
  theme_minimal(base_size = 14) +
  labs(title = "True vs Predicted Class Counts",
       x = "Class",
       y = "Count")

##########################################
# ---- PLOT D: Prediction Probabilities ---
##########################################

probs$Patient <- results$Patient


probs_long <- probs %>%
  pivot_longer(cols = c(HELA, MS751, `OMC-1`),
               names_to = "Class",
               values_to = "Probability")


plotD <- ggplot(probs_long, aes(x = Class, y = Probability)) +
  geom_boxplot(fill = "steelblue", alpha = 0.8) +
  theme_minimal(base_size = 14) +
```

```
  labs(title = "Prediction Probability Distribution",
       x = "Class",
       y = "Probability")

##############################################
# -------- Combine as Multi-Panel ---------
##############################################
multi_fig <- (plotA | plotB) / (plotC | plotD) +
  plot_annotation(tag_levels = "A")


##############################################
# ---------- Export 600 DPI TIFF ----------
##############################################
tiff(
  "RF_MultiPanel_Figure.tiff",
  width = 4200,
  height = 3600,
  res = 600,
  compression = "lzw"
)


print(multi_fig)
dev.off()
```



CONCLUSIONS

This handbook demonstrates that integrating tumour microenvironment (TME)–aware epigenomic features into patient stratification yields biologically coherent and clinically relevant cervical cancer endotypes. By explicitly disentangling microenvironment-associated regulatory programmes from tumour-intrinsic oncogenic processes and global epigenetic alterations, the framework captures critical axes of tumour heterogeneity that are routinely obscured in conventional genome-wide analyses. The identification of immune-hot and immune-cold methylation endotypes, together with their distinct clinical and virological associations, highlights the central role of the tumour microenvironment in shaping disease progression and therapeutic vulnerability. Crucially, this work moves beyond descriptive tumour stratification by directly linking patient-defined epigenetic states to experimentally tractable cancer models. Systematic tumour–model concordance analyses demonstrate that only a subset of widely used cervical cancer experimental systems faithfully recapitulate patient-specific epigenetic and microenvironmental programmes, exposing a major and often under-recognised source of irreproducibility in preclinical research. By integrating complementary epigenetic feature spaces within a unified machine-learning framework, the approach provides a principled and scalable strategy for experimental model recommendation, replacing ad hoc selection with tumour-informed inference. Collectively, these findings establish a robust framework for tumour-aware precision modelling. By embedding microenvironmental context directly into epigenomic analysis and experimental model prioritisation, the framework enhances the biological validity and translational relevance of preclinical studies and offers a generalisable strategy for improving model fidelity and therapeutic discovery across cancers, particularly in disease settings characterised by complex tumour–immune–viral interactions.