Introduction to OmniAgeR

Zhaozhen Du[1*] ***and Andrew E. Teschendorff***[1**]

[1]Shanghai Institute of Nutrition and Health, CAS

[*]duzhaozhen2022@sinh.ac.cn (mailto:duzhaozhen2022@sinh.ac.cn)
[**]andrew@sinh.ac.cn (mailto:andrew@sinh.ac.cn)

1 February 2026

Package

OmniAgeR 0.7.0

Contents

1 Overview of Implemented Clocks

The `OmniAgeR` package provides two primary, high-level interfaces: EpiAge(): For calculating a comprehensive suite of aging-related clocks (chronological, biological, mitotic, etc.). EpiGA(): For calculating a specialized suite of gestational age clocks. For advanced users or specific applications, the package also provides direct access to individual clock functions (e.g., Horvath2013()) and specialized "bundle" calculators (e.g., PCClocks(), SystemsAge()) that manage complex dependencies. These tools are broadly categorized into three main groups:

1. **Epigenetic (DNAm) Clocks**
2. **Transcriptomic (RNA) Clocks**
3. **Other Aging-Related Surrogate Biomarkers**

The suite of **epigenetic aging clocks** is particularly extensive, organized into several functional categories, including predictors for chronological age, biological age, cellular division (mitotic clocks), and gestational age.

2 Epigenetic (DNAm) Aging Clocks

2.1 Chronological Age Clocks

These DNAm clocks are designed for the prediction of chronological age.

- `Horvath2013` : (Horvath 2013) The original pan-tissue clock from 353 CpGs.
- `Hannum` : (Hannum et al. 2013) Blood-specific clock from 71 CpGs.
- `Lin` : (Lin et al. 2016) Clock based on 99 CpGs.
- `VidalBralo` : (Vidal-Bralo et al. 2016) Clock based on 8 CpGs.
- `ZhangClock` : (Zhang et al. 2019) Improved-precision clock from 514 CpGs.
- `Horvath2018` : (Horvath et al. 2018) Skin & blood clock.
- `Bernabeu_cAge` : (Bernabeu et al. 2022) Clock based on 3225 CpGs.
- `CorticalClock` : (Shireby et al. 2020) Brain cortical clock based on 347 CpGs.
- `PedBE` : (McEwen et al. 2019) Pediatric buccal epithelial clock for children.
- `CentenarianClock` : (Eric Dec et al. 2023) Centenarian Epigenetic Clocks.
- `Retro_age` : (Ndhlovu et al. 2024) Retroelement-based clock developed on the EPIC v1/v2 arrays.
- `PCHorvath2013` , `PCHorvath2018` , `PCHannum` : (Higgins-Chen et al. 2022) Computationally-bolstered versions of the original clocks. These "PC clocks" are retrained using principal components (PCs) derived from a large CpG set to minimize technical noise and improve reliability

2.2 Biological Age Clocks

These advanced clocks are designed to capture biological aging rather than chronological time, often showing stronger associations with health outcomes and mortality.

- `Zhang10` : (Zhang et al. 2017) 10-CpG clock associated with mortality.
- `PhenoAge` : (Levine et al. 2018) Predicts phenotypic age from 513 CpGs.
- `DunedinPACE` : (Belsky et al. 2022) Quantifies the pace of biological aging.
- `GrimAge1` : The original GrimAge clock (Lu et al. 2019).
- `GrimAge2` : (Lu et al. 2022) Updated composite biomarker of mortality risk.
- `PCPhenoAge` , `PCGrimAge1` : (Higgins-Chen et al. 2022) Computationally-bolstered PC versions of the PhenoAge and GrimAge1 clocks, retrained on principal components for enhanced reliability.
- `DNAmFitAge` : (McGreevy et al. 2023) Biological age indicator incorporating DNAmGrimAge and 3 DNAm-based physical fitness markers.
- `IC_Clock` : (Fuentealba et al. 2025) The Intrinsic Capacity (IC) Clock based on 91 CpGs.
- `SystemsAge` : (Sehgal et al. 2025) The Systems Age and 11 System-Specific Scores.

2.3 Cellular Aging Clocks

This category groups biomarkers that measure two key forms of cellular aging: cell proliferation (mitotic clocks) and telomere attrition (DNAmTL).

2.3.1 Mitotic Clocks (Cellular Division)

A specialized set of clocks that measure cell division history or stem cell divisions.

- `epiTOC1` : (Yang et al. 2016) Average beta value of 385 promoter CpGs.
- `epiTOC2` : (Teschendorff et al. 2020) Estimates stem cell divisions using a dynamic model.
- `epiTOC3` : Estimates stem cell divisions based on unmethylated population doubling associated CpGs.

- `stemTOCvitro` : (Zhu et al. 2024) The 0.95 upper quantile of 629 stemTOCvitro CpGs (promoter CpGs unmethylated in fetal tissue that hypermethylate with population-doublings).
- `stemTOC` : (Zhu et al. 2024) The 0.95 upper quantile of 371 stemTOC CpGs, filtered for in-vivo hypermethylation with age.
- `RepliTali` : (Endicott et al. 2022) Based on 87 population doubling associated hypomethylated CpGs.
- `HypoClock` : (Teschendorff et al. 2020) Based on hypomethylation at 678 solo-WCGW sites.
- `EpiCMIT_hyper` : (Duran-Ferrer et al. 2020) Average beta value of 184 age-associated hypermethylated CpGs.
- `EpiCMIT_hypo` : (Duran-Ferrer et al. 2020) Average beta value of 1164 age-associated hypomethylated CpGs.

2.3.2 DNAm Telomere Length (TL) Clocks

- `DNAmTL` : (Lu AT et al. 2019) Calculates the Leukocyte telomere length.
- `PCDNAmTL` : (Lu AT et al. 2019; Higgins-Chen et al. 2022) A computationally-bolstered "PC" version of the original DNAmTL clock. This clock is retrained using principal components (PCs) derived from a large CpG set to minimize technical noise and improve reliability.

2.4 Causal Clocks

A new generation of clocks developed to reflect potential causal drivers of aging.

- `CausalAge` , `DamAge` , `AdaptAge` : (Ying et al. 2024) Three clocks derived from a causality-enriched model. They dissect aging into distinct components: 'Causal' (CausalAge), 'Damage' (DamAge), and Adaptation' (AdaptAge).

2.5 Stochastic Clocks

- `StocH` , `StocP` , `StocZ` : (Tong et al. 2024) Stochastic analogues of the Horvath, PhenoAge, and Zhang clocks, trained on artificial cohorts to quantify the stochastic component of epigenetic aging.

2.6 Cell-Type Specific Clocks

These clocks are designed to measure aging in specific cell types or tissues, accounting for cellular heterogeneity.

- `Neu-In` , `Glia-In` : (Tong et al. 2024) Intrinsic clocks for neurons, and glia.
- `Neu-Sin` , `Glia-Sin` , `Hep` : (Tong et al. 2024) Semi-intrinsic clocks for neurons, glia, and hepatocytes

2.7 Gestational Age Clocks

This collection includes DNAm clocks designed to predict gestational age at birth.

- `Bohlin_GA` : The Bohlin Gestational Age.
- `EPIC_GA` : The EPIC Gestational Age.
- `Lee_GA` : A list (LeeControl, LeeRobust, LeeRefinedRobust).
- `Knight_GA` : The Knight Gestational Age.
- `Mayne_GA` : The Mayne Placental Gestational Age.

2.8 Cross-Species Support

- `EnsembleAgeMouse_Dynamic` , `EnsembleAgeMouse_Static` , `EnsembleAgeHumanMouse` : (Haghani et al. 2025) The multi-clock framework implementation. This includes mouse-specific estimators (Static and Dynamic) and a cross-species (HumanMouse) predictor applicable to both humans and mice

- `UniversalPanMammalianClocks` : (Lu et al. 2023) The three pan-tissue clocks (Clock 1, 2, 3) applicable across all mammalian tissues.

- `PanMammalianBlood` : (Lu et al. 2023) The two blood-specific pan-mammalian clocks (Clock 2, 3).

- `PanMammalianSkin` : (Lu et al. 2023) The two skin-specific pan-mammalian clocks (Clock 2, 3).

3 Transcriptomic Clocks

In addition to DNAm-based models, the package implements cutting-edge transcriptomic (RNA-seq) clocks:

- `sc-ImmuAging` : A single-cell, immune-cell-type-specific aging clock.
- `Brain_CT_clock` : A brain-cell-type-specific aging clock based on Single-Nuclei Transcriptomics.
- `Pasta` : A robust, multi-tissue transcriptomic clock based on a novel age-shift learning strategy. It utilizes rank-normalization to ensure broad applicability across heterogeneous datasets, including bulk RNA-seq, single-cell pseudobulks, and microarrays.

4 Surrogate Biomarkers & Scores

Besides, `OmniAgeR` includes functions to compute several other important surrogate markers associated with aging, lifestyle, and health status from DNAm data.

- `CompCRP` : Calculates a score for C-reactive protein (CRP) levels, a marker of inflammation.
- `CompCHIP` : Calculates scores for Clonal Hematopoiesis of Indeterminate Potential (CHIP).
- `CompIL6` : Computes a DNA methylation (DNAm) surrogate score for Interleukin-6 (IL-6) protein levels.
- `CompEpiScores` : Computes the 109 validated epigenetic scores (EpiScores) that serve as DNA methylation-based proxies for the levels of circulating plasma proteins as defined by Gadd et al. (2022).
- `McCartney_Trait` : Computes epigenetic surrogate scores for 10 different complex traits and biomarkers based on blood methylation models. Traits include: `BMI` , `Smoking` (pack-years), `Alcohol` (units/week), `Education` (years), `Total_cholesterol` , `HDL_cholesterol` , `LDL_cholesterol` , `Total_HDL_ratio` , `WHR` (Waist-to-Hip Ratio), and `Body_fat_Perc` .

5 DNA Methylation Cell-Type Fraction

The OmniAgeR package also provides a DNA Methylation Cell-Type Fraction (CTF) clock (implemented as `DNAm_CTF_Clock`), which was trained on a large-scale NSPT dataset comprising more than 3,500 blood samples. The clock incorporates 12 immune cell types, estimated using the EpiDISH algorithm, to model aging based on cell-type composition.

6 Available Clocks and Functions

6.1 Epigenetic (DNAm) Aging Clocks

Clock Name	Category	Calculator / Function
`Horvath2013`	Chronological	`EpiAge()` , `Horvath2013()`
`Hannum`	Chronological	`EpiAge()` , `Hannum()`
`Lin`	Chronological	`EpiAge()` , `Lin()`
`VidalBralo`	Chronological	`EpiAge()` , `VidalBralo()`
`ZhangClock`	Chronological	`EpiAge()` , `ZhangClock()`
`Horvath2018`	Chronological	`EpiAge()` , `Horvath2018()`
`Bernabeu_cAge`	Chronological	`EpiAge()` , `Bernabeu_cAge()`
`CorticalClock`	Chronological	`EpiAge()` , `CorticalClock()`

Clock Name	Category	Calculator / Function
`PedBE`	Chronological	`EpiAge()` , `PedBE()`
`CentenarianClock`	Chronological	`EpiAge()` , `CentenarianClock`
`Retro_age`	Chronological	`EpiAge()` , `Retro_age()`
`Zhang10`	Biological	`EpiAge()` , `Zhang10()`
`PhenoAge`	Biological	`EpiAge()` , `PhenoAge()`
`DunedinPACE`	Biological	`EpiAge()` , `DunedinPACE()`
`GrimAge1`	Biological	`EpiAge()` , `GrimAge1()`
`GrimAge2`	Biological	`EpiAge()` , `GrimAge2()`
`DNAmFitAge`	Biological	`EpiAge()` , `DNAmFitAge()`
`IC_Clock`	Biological	`EpiAge()` , `IC_Clock()`
`SystemsAge`	Biological	`EpiAge()` , `SystemsAge()`
`epiTOC1`	Mitotic	`EpiAge()` , `epiTOC1()`
`epiTOC2`	Mitotic	`EpiAge()` , `epiTOC2()`
`epiTOC3`	Mitotic	`EpiAge()` , `epiTOC3()`
`stemTOCvitro`	Mitotic	`EpiAge()` , `stemTOCvitro()`
`RepliTali`	Mitotic	`EpiAge()` , `RepliTali()`
`HypoClock`	Mitotic	`EpiAge()` , `HypoClock()`
`EpiCMIT_hyper` , `EpiCMIT_hypo`	Mitotic	`EpiAge()` , `EpiCMIT()`
`DNAmTL`	Telomere length	`EpiAge()` , `DNAmTL()`
`PCHorvath2013` , `PCHorvath2018` , `PCHannum` , `PCPhenoAge` , `PCGrimAge1` , `PCDNAmTL`	Chronological/Biological/Telomere length	`EpiAge()` , `PCClocks()`
`CausalAge` , `DamAge` , `AdaptAge`	Causal	`EpiAge()` , `CausalClock()`
`StocH` , `StocP` , `StocZ`	Causal	`EpiAge()` , `StochClocks()`
`Neu-In` , `Glia-In` , `Neu-Sin` , `Glia-Sin` , `Hep`	Cell-Type Specific	`EpiAge()` , `CTS_Clocks()`
`Bohlin_GA`	Gestational	`EpiGA()` , `Bohlin_GA()`
`EPIC_GA`	Gestational	`EpiGA()` , `EPIC_GA()`
`Lee_GA`	Gestational	`EpiGA()` , `Lee_GA()`
`Knight_GA`	Gestational	`EpiGA()` , `Knight_GA()`
`Mayne_GA`	Gestational	`EpiGA()` , `Mayne_GA()`
`EnsembleAge`	Mouse & Human	`EnsembleAge()`
`PanMammalianClocks`	PanMammalian	`UniversalPanMammalianClock`
`PanMammalianBlood`	PanMammalian	`PanMammalianBlood()`
`PanMammalianSkin`	PanMammalian	`PanMammalianSkin()`
`DNAm_CTF_Clock`	Cell-Type Fraction Clock	`DNAm_CTF_Clock()`

6.2 Transcriptomic Clocks

Clock Name	Category	Calculator / Function
`sc-ImmuAging`	Immune-cell-type-specific	`scImmuAging()`
`Brain_CT_clock`	Brain-cell-type-specific	`Brain_CT_clock()`

Clock Name	Category	Calculator / Function
`Pasta`	Multi-tissue	`PASTA_Scores()`

6.3 Surrogate Biomarkers & Scores

Clock Name	Target	Calculator / Function
`CompCRP`	CRP/intCRP score	`CompCRP()`
`CompCHIP`	CHIP score	`CompCHIP()`
`CompIL6`	IL6 score	`CompIL6()`
`CompEpiScores`	109 validated epigenetic scores	`CompEpiScores()`
`McCartney_Trait`	Epigenetic surrogate scores for 10 different complex traits	`McCartney_Trait()`

7 Tutorial Example 1: Apply mitotic clocks on Lung pre-cancerous lesions

7.1 Loading the lung pre-cancerous lesions dataset

This is an Illumina 450k dataset encompassing 21 normal lung tissue samples and 35 lung carcinoma in situ (LCIS) samples, of which 22 progressed to an invasive lung cancer (LC). Here we explore the correlation between mitotic age and cancer state with linear model, treating N, LCIS, and LC as ordinal variable (1,2,3) and adjusting for age.

```
library(OmniAgeR)
```

```
## Setting options('download.file.method.GEOquery'='auto')
```

```
## Setting options('GEOquery.inmemory.gpl'=FALSE)
```

```
library(ggplot2)
library(patchwork)
library(ggpubr)
download_OmniAgeR_example("LungInv")
```

```
## All requested example datasets already exist.
```

```
load_OmniAgeR_example("LungInv")
```

```
## Dataset 'LungInv' loaded into environment.
```

```
my_comparisons <- list(c("N\nN=21", "LCIS\nN=13"), c("LCIS\nN=13", "LCIS->LC\nN=2
table(df$Group)
```

```
##
##     LCIS\nN=13 LCIS->LC\nN=22       N\nN=21
##             13             22             21
```

```
## Check available epigenetic clocks
listEpiAge()
```

```
## $mitotic
## [1] "epiTOC1"       "epiTOC2"       "epiTOC3"       "stemTOCvitro"
## [5] "stemTOC"       "RepliTali"     "HypoClock"     "EpiCMIT_Hyper"
## [9] "EpiCMIT_Hypo"
##
## $dnamtl
## [1] "DNAmTL"   "PCDNAmTL"
##
## $chronological
##  [1] "Horvath2013"      "Hannum"           "Lin"              "VidalBralo"
##  [5] "ZhangClock"       "Horvath2018"      "Bernabeu_cAge"    "CorticalClock"
##  [9] "PedBE"            "CentenarianClock" "Retro_age"        "PCHorvath2013"
## [13] "PCHorvath2018"    "PCHannum"
##
## $biological
##  [1] "Zhang10"     "PhenoAge"    "DunedinPACE" "GrimAge1"    "GrimAge2"
##  [6] "PCPhenoAge"  "PCGrimAge1"  "DNAmFitAge"  "IC_Clock"    "SystemsAge"
##
## $causal
## [1] "CausalAge" "DamAge"    "AdaptAge"
##
## $celltype_specific
## [1] "Neu-In"   "Neu-Sin"  "Glia-In"  "Glia-Sin" "Hep"
```

7.2 Estimation of epigenetic mitotic age

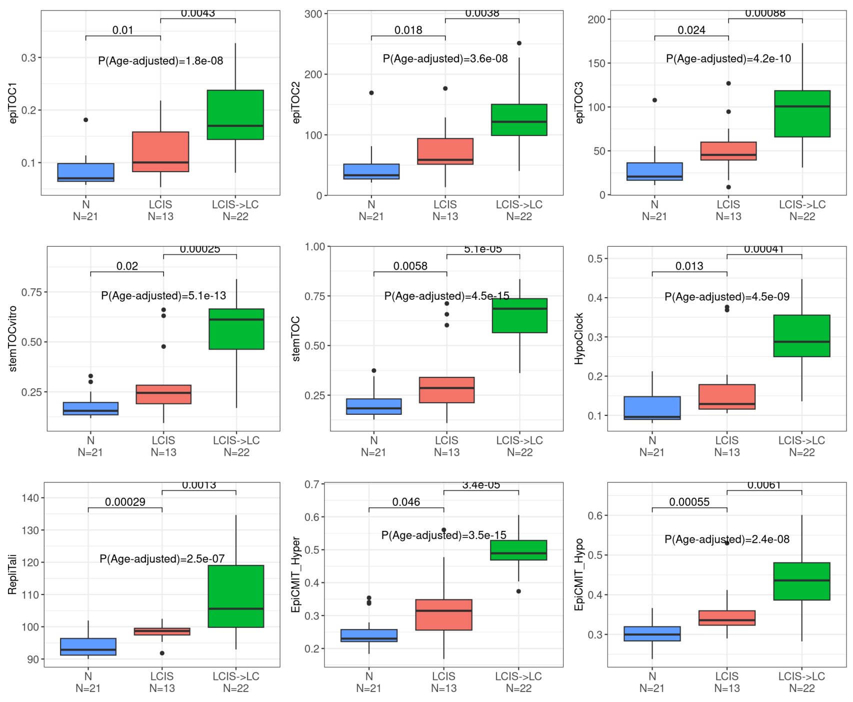
Mitotic age estimation of all the clocks is implemented in a general function *EpiMitClocks*. The required input is a DNAm beta value matrix with rows labeling Illumina 450k/EPIC CpGs and columns labeling samples.

```
EpiAge.o<-EpiAge(data.m = bmiq.m,clock_names = "mitotic",ages.v = df$Age)
```

```
## [1] "[epiTOC1] Number of represented epiTOC1 CpGs (max=385)=379"
## [1] "[epiTOC2] Number of represented epiTOC2 CpGs (max=163)=161"
## [1] "[epiTOC3] Number of represented epiTOC3 CpGs (max=170)=164"
## [1] "[stemTOCvitro] Number of represented stemTOCvitro CpGs (max=629)=615"
## [1] "[stemTOC] Number of represented stemTOC CpGs (max=371)=360"
## [1] "[HypoClock] Number of represented solo-WCGWs (max=678)=672"
## [1] "[RepliTali] Number of represented RepliTali CpGs (max=87)=30"
## [1] "[EpiCMIT] Number of represented EpiCMIT_hyper CpGs (max=184)=182"
## [1] "[EpiCMIT] Number of represented EpiCMIT_hypo CpGs (max=1164)=1158"
```

```
rm(bmiq.m)
```

```
df$epiTOC1 <-EpiAge.o$epiTOC1
df$epiTOC2 <-EpiAge.o$epiTOC2$irS
df$epiTOC3 <-EpiAge.o$epiTOC3$irS
df$stemTOCvitro <-EpiAge.o$stemTOCvitro
df$stemTOC<-EpiAge.o$stemTOC
df$HypoClock<-EpiAge.o$HypoClock
df$RepliTali<-EpiAge.o$RepliTali
df$EpiCMIT_Hyper<-EpiAge.o$EpiCMIT_Hyper
df$EpiCMIT_Hypo<-EpiAge.o$EpiCMIT_Hypo
g<-list()
for (i in 1:9){
  temp.df<-data.frame('Group'=df$Group,'Age'=df$Age,'num'=df$num,'score'=df[,i+3]
  y<-summary(lm(temp.df$score~temp.df$num+temp.df$Age))$coefficients[2,4]
  g[[i]]<-ggplot(temp.df,aes(x=Group,y=score,fill=Group))+guides(fill='none')+
  geom_boxplot()+theme_bw()+
  scale_x_discrete(limits=c('N\nN=21','LCIS\nN=13','LCIS->LC\nN=22'))+
  stat_compare_means(method='wilcox.test',comparisons = my_comparisons, size = 4)
  xlab('')+ylab(colnames(df)[i+3])+
  annotate('text',x=2,y=max(temp.df$score) * 0.9,label=paste0('P(Age-adjusted)=',
  theme(
      axis.title = element_text(size = 11),
      axis.text = element_text(size = 11)
    )

}

ggpubr::ggarrange(plotlist = g, nrow = 3, ncol = 3)
```



8 Tutorial Example 2: Predict chronological age in blood tissue

The epigenetic age of 50 normal blood samples was estimated from their Illumina 450k methylation profiles using both the Horvath and Zhang epigenetic clocks.

```
library(OmniAgeR)
library(ggplot2)
library(patchwork)
library(ggpubr)
download_OmniAgeR_example("Hannum_example")
```

```
## All requested example datasets already exist.
```

```
load_OmniAgeR_example("Hannum_example")
```

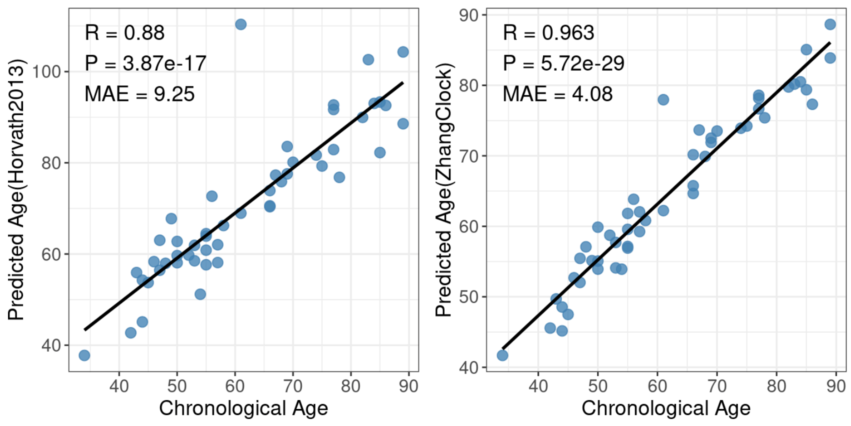
```
## Dataset 'Hannum_example' loaded into environment.
```

```
age <- PhenoTypesHannum_lv$Age
sex <- ifelse(PhenoTypesHannum_lv$Sex=="F","Female","Male")
EpiAge.out <- EpiAge(hannum_bmiq_m,clock_names = c("Horvath2013","ZhangClock"))
```

```
## [1] "[Horvath2013] Number of represented Horvath2013 CpGs (max=353)=353"
## [1] "[ZhangClock] Number of represented Zhang CpGs (max=514)=513"
```

```
gp<-list()

for (i in seq_along(EpiAge.out)){
  plot_df <- data.frame(ActualAge = PhenoTypesHannum_lv$Age, PredictedAge = EpiAg
  cor_test_result <- cor.test(plot_df$ActualAge, plot_df$PredictedAge)
  correlation <- cor_test_result$estimate
  p_value <- cor_test_result$p.value
  mae <- mean(abs(plot_df$PredictedAge - plot_df$ActualAge))
  p_value_formatted <- ifelse(p_value < 0.001,
                              formatC(p_value, format = "e", digits = 2),
                              round(p_value, 3))
  annotation_text <- paste0(
  "R = ", round(correlation, 3), "\n",
  "P = ", p_value_formatted, "\n",
  "MAE = ", round(mae, 2)
  )
  gp[[i]]<-ggplot(data = plot_df, aes(x = ActualAge, y = PredictedAge)) +
  geom_point(color = "steelblue", size = 3, alpha = 0.8) +
  geom_smooth(method = "lm",color = "black", se = FALSE) +
  annotate("text",
           x = min(plot_df$ActualAge, na.rm = TRUE),
           y = max(plot_df$PredictedAge, na.rm = TRUE),
           label = annotation_text,
           hjust = 0,
           vjust = 1,             size = 5) +

  labs(
    title = NULL,
    x = "Chronological Age",
    y = paste0("Predicted Age(", c("Horvath2013","ZhangClock")[i],")")
  ) +
  theme_bw() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, size = 12),
    axis.title = element_text(size = 14),
    axis.text = element_text(size = 12)
  )

}

ggpubr::ggarrange(plotlist = gp, nrow = 1, ncol = 2)
```

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



9 Tutorial Example 3: Predict gestational age

A small demonstration dataset containing three samples, used to illustrate how to predict the gestational age of samples using DNAm Gestational Age Clocks.

```
library(OmniAgeR)
library(ggplot2)
library(patchwork)
library(ggpubr)
## Check available epigenetic clocks
listEpiGA()
```

```
## [1] "Bohlin_GA" "EPIC_GA"   "Knight_GA" "Lee_GA"    "Mayne_GA"
```

```
download_OmniAgeR_example("GA_example")
```

```
## All requested example datasets already exist.
```

```
load_OmniAgeR_example("GA_example")
```

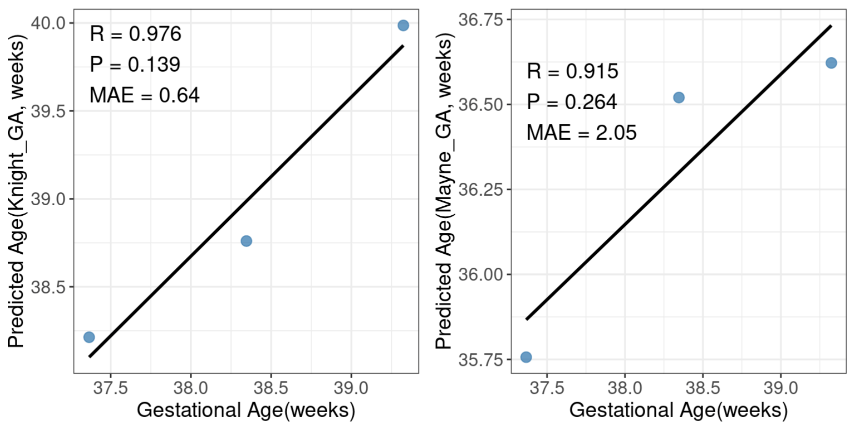
```
## Dataset 'GA_example' loaded into environment.
```

```
EpiGA.o <- EpiGA(GA_m,clock_names = c("Knight_GA","Mayne_GA"))
```

```
## [1] "[Knight_GA] Number of represented Knight_GA CpGs (max=148)=148"
## [1] "[Mayne_GA] Number of represented Mayne_GA CpGs (max=62)=62"
```

```
gp<-list()

for (i in seq_along(EpiGA.o)){
  plot_df <- data.frame(Gestational_Age = phenotypeGA$age, PredictedAge = EpiGA.c
  cor_test_result <- cor.test(plot_df$Gestational_Age, plot_df$PredictedAge)
  correlation <- cor_test_result$estimate
  p_value <- cor_test_result$p.value
  mae <- mean(abs(plot_df$PredictedAge - plot_df$Gestational_Age))
  p_value_formatted <- ifelse(p_value < 0.001,
                              formatC(p_value, format = "e", digits = 2),
                              round(p_value, 3))
  annotation_text <- paste0(
  "R = ", round(correlation, 3), "\n",
  "P = ", p_value_formatted, "\n",
  "MAE = ", round(mae, 2)
  )
  gp[[i]]<-ggplot(data = plot_df, aes(x = Gestational_Age, y = PredictedAge)) +
  geom_point(color = "steelblue", size = 3, alpha = 0.8) +
  geom_smooth(method = "lm",color = "black", se = FALSE) +
  annotate("text",
           x = min(plot_df$Gestational_Age, na.rm = TRUE),
           y = max(plot_df$PredictedAge, na.rm = TRUE),
           label = annotation_text,
           hjust = 0,
           vjust = 1,             size = 5) +

  labs(
    title = NULL,
    x = "Gestational Age(weeks)",
    y = paste0("Predicted Age(", c("Knight_GA","Mayne_GA")[i],", weeks)")
  ) +
  theme_bw() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, size = 12),
    axis.title = element_text(size = 14),
    axis.text = element_text(size = 12)
  )

}

ggpubr::ggarrange(plotlist = gp, nrow = 1, ncol = 2)
```

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



10 Tutorial Example 4: Apply sc-ImmuAging to PBMC scRNA-seq dataset

In this example, we showcase the application of the cell-type-specific clock, scImmuAging, to a PBMC scRNA-seq dataset. The dataset includes 20 samples, profiled to contain only CD4+ and CD8+ T cells.

```
library(OmniAgeR)
library(Seurat)
```

```
## Loading required package: SeuratObject
```

```
## Loading required package: sp
```

```
##
## Attaching package: 'SeuratObject'
```

```
## The following objects are masked from 'package:base':
##
##     intersect, t
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-10
```

```
library(ggplot2)
library(patchwork)
library(ggpubr)
download_OmniAgeR_example("Yazar_CD4T_CD8T_example")
```

```
## All requested example datasets already exist.
```

```
load_OmniAgeR_example("Yazar_CD4T_CD8T_example")
```

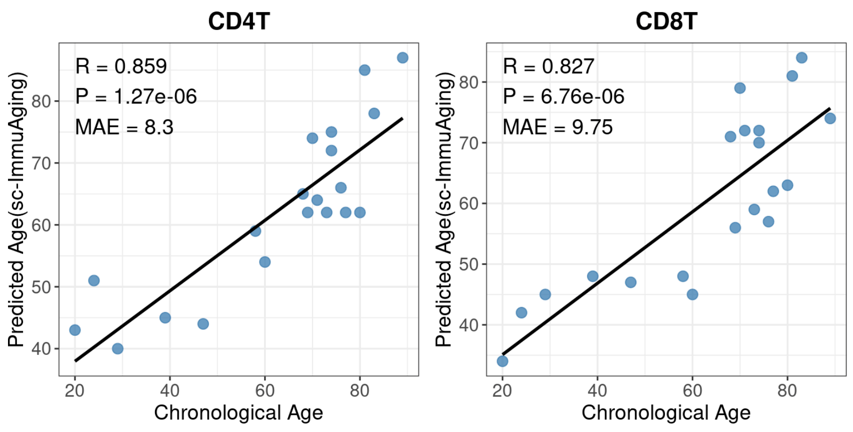
```
## Dataset 'Yazar_CD4T_CD8T_example' loaded into environment.
```

```
scImmuAging.out <- scImmuAging(seurat_obj,c("CD4T","CD8T"))
```

```
## Using scImmuAging to predict age
##
## --- Processing cell type: CD4T ---
## Selecting model and features for: CD4T
## Pre-processing data and generating pseudocells...
## Calculating age predictions...
## Aggregating predictions for each donor...
## Prediction complete for CD4T !
##
## --- Processing cell type: CD8T ---
## Selecting model and features for: CD8T
## Pre-processing data and generating pseudocells...
## Calculating age predictions...
## Aggregating predictions for each donor...
## Prediction complete for CD8T !
##
## --- All predictions complete! ---
```

```
sc_gp<-list()

for (i in seq_along(scImmuAging.out)){
  plot_df <- data.frame(ActualAge = scImmuAging.out[[i]]$Donor$age, PredictedAge
  cor_test_result <- cor.test(plot_df$ActualAge, plot_df$PredictedAge)
  correlation <- cor_test_result$estimate
  p_value <- cor_test_result$p.value
  mae <- mean(abs(plot_df$PredictedAge - plot_df$ActualAge))
  p_value_formatted <- ifelse(p_value < 0.001,
                              formatC(p_value, format = "e", digits = 2),
                              round(p_value, 3))
  annotation_text <- paste0(
  "R = ", round(correlation, 3), "\n",
  "P = ", p_value_formatted, "\n",
  "MAE = ", round(mae, 2)
  )
  sc_gp[[i]]<-ggplot(data = plot_df, aes(x = ActualAge, y = PredictedAge)) +
  geom_point(color = "steelblue", size = 3, alpha = 0.8) +
  geom_smooth(method = "lm",color = "black", se = FALSE) +
  annotate("text",
           x = min(plot_df$ActualAge, na.rm = TRUE),
           y = max(plot_df$PredictedAge, na.rm = TRUE),
           label = annotation_text,
           hjust = 0,
           vjust = 1,
           size = 5) +

  labs(
    title = c("CD4T","CD8T")[i],
    x = "Chronological Age",
    y = "Predicted Age(sc-ImmuAging)"
  ) +
  theme_bw() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, size = 12),
    axis.title = element_text(size = 14),
    axis.text = element_text(size = 12)
  )

}

ggpubr::ggarrange(plotlist = sc_gp, nrow = 1, ncol = 2)
```

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



11 Tutorial Example 5: Apply Brain_CT_clock to Brain snRNA-seq dataset

In this example, we showcase the application of the brain cell-type-specific aging Clocks to a brain snRNA-seq dataset. The dataset includes 15 donors, profiled to contain only Oligodendrocytes.

```
library(OmniAgeR)
library(Seurat)
library(glmnet)
library(ggplot2)
library(patchwork)
library(ggpubr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
download_OmniAgeR_example("brain_frohlich_control_example_15donors")
```

```
## All requested example datasets already exist.
```

```
load_OmniAgeR_example("brain_frohlich_control_example_15donors")
```

```
## Dataset 'brain_frohlich_control_example_15donors' loaded into environment.
```

```
brain_clock_results <- Brain_CT_clock(
   seurat_object = brain_seurat,
   cell_types = c("Oligodendrocytes"),
   model_name = "all" ##  c('SC', 'Pseudobulk', 'Bootstrap')
 )
```

```
## Starting Brain_CT_clock for cell types
## [1] "--- Starting SC Model Pipeline ---"
## [1] "Processing SC Model for: Oligodendrocytes"
```

```
## Warning in asMethod(object): sparse->dense coercion: allocating vector of size
## 2.5 GiB
```

```
## [1] "--- Finished SC Model Pipeline ---"
## [1] "--- Starting Pseudobulk Model Pipeline ---"
## [1] "Processing Pseudobulk Model for: Oligodendrocytes"
```

```
## Warning in asMethod(object): sparse->dense coercion: allocating vector of size
## 2.5 GiB
```

```
## [1] "--- Finished Pseudobulk Model Pipeline ---"
## [1] "--- Starting Bootstrap Model Pipeline ---"
## [1] "Processing Bootstrap Model for: Oligodendrocytes"
```

```
## Warning in asMethod(object): sparse->dense coercion: allocating vector of size
## 2.5 GiB
```

```
## [1] "--- Finished Bootstrap Model Pipeline ---"
##
## --- All Brain_CT_clock predictions complete! ---
```

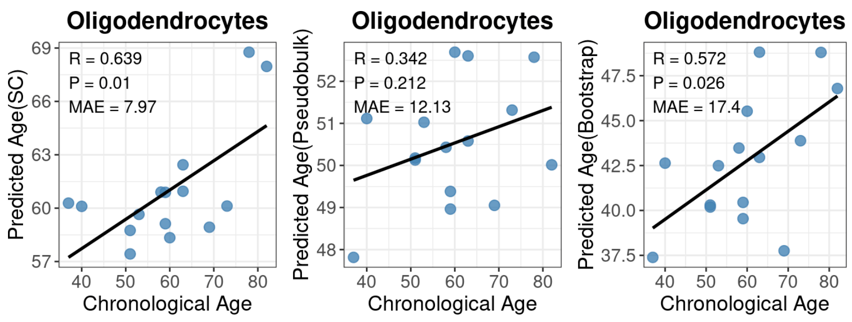
```
# Use lapply to iterate over each data frame in the list
average_predictions_by_donor <- lapply(brain_clock_results, function(df) {
  df %>%
    group_by(donors,ages,sample_type,celltype) %>%
    summarise(mean_prediction = mean(predictions, na.rm = TRUE))
})
```

```
## `summarise()` has grouped output by 'donors', 'ages', 'sample_type'. You can
## override using the `.groups` argument.
```

```
## `summarise()` has grouped output by 'donors', 'ages', 'sample_type'. You can
## override using the `.groups` argument.
## `summarise()` has grouped output by 'donors', 'ages', 'sample_type'. You can
## override using the `.groups` argument.
```

```
brain_gp<-list()

for (i in seq_along(average_predictions_by_donor)){

  plot_df <- data.frame(ActualAge = average_predictions_by_donor[[i]]$ages, Predi
  cor_test_result <- cor.test(plot_df$ActualAge, plot_df$PredictedAge)
  correlation <- cor_test_result$estimate
  p_value <- cor_test_result$p.value
  mae <- mean(abs(plot_df$PredictedAge - plot_df$ActualAge))
  p_value_formatted <- ifelse(p_value < 0.001,
                          formatC(p_value, format = "e", digits = 2),
                          round(p_value, 3))
  annotation_text <- paste0(
  "R = ", round(correlation, 3), "\n",
  "P = ", p_value_formatted, "\n",
  "MAE = ", round(mae, 2)
  )
  brain_gp[[i]]<-ggplot(data = plot_df, aes(x = ActualAge, y = PredictedAge)) +
  geom_point(color = "steelblue", size = 3, alpha = 0.8) +
  geom_smooth(method = "lm",color = "black", se = FALSE) +
  annotate("text",
           x = min(plot_df$ActualAge, na.rm = TRUE),
           y = max(plot_df$PredictedAge, na.rm = TRUE),
           label = annotation_text,
           hjust = 0,
           vjust = 1,
           size = 4) +

  labs(
    title = "Oligodendrocytes",
    x = "Chronological Age",
    y = paste0("Predicted Age(",c('SC', 'Pseudobulk', 'Bootstrap')[i],")")
  ) +
  theme_bw() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, size = 12),
    axis.title = element_text(size = 14),
    axis.text = element_text(size = 12)
  )

}

ggpubr::ggarrange(plotlist = brain_gp, nrow = 1, ncol = 3)
```

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



12 Tutorial Example 6: Apply Pasta to Brain (MTG) scRNA-seq dataset

In this example, we showcase the application of the Pasta clock to a single-nuclei RNA-seq dataset from the middle temporal gyrus (MTG) (Gabitto et al., 2024). The dataset includes 46 healthy donors and is filtered to contain extratelencephalic projecting glutamatergic cortical neurons.

```r
# Load required libraries
library(OmniAgeR)
library(Seurat)
library(glmnet)
library(magrittr) # For %>% pipe
library(ggplot2)
library(patchwork)
library(ggpubr)    # For simplified plot annotations
# Load the pre-filtered Seurat object
download_OmniAgeR_example("seu_gabitto_2024_filtered")
```

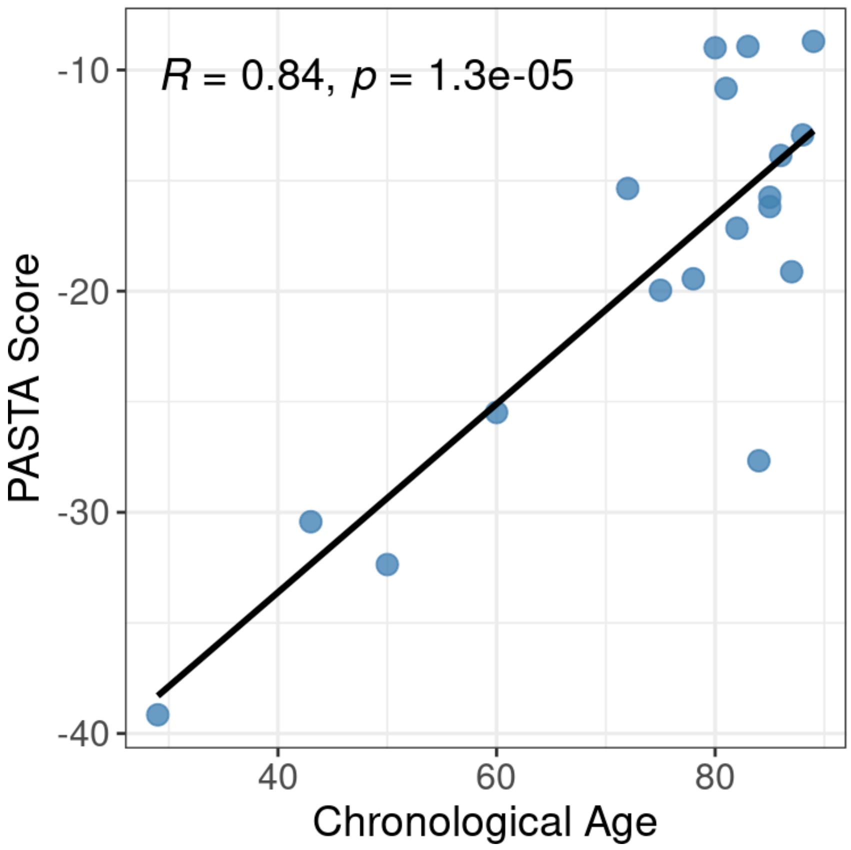
```
## All requested example datasets already exist.
```

```r
load_OmniAgeR_example("seu_gabitto_2024_filtered")
```

```
## Dataset 'seu_gabitto_2024_filtered' loaded into environment.
```

```r
# Extract and clean chronological age from metadata
seu$age <- seu$development_stage %>%
  gsub("-year.*", "", .) %>%             # Remove suffixes
  gsub("-", " ", .) %>%                 # Remove hyphens
  gsub("80 year old and over stage", "85", .) # Standardize 80+ category

set.seed(42)

# Create pseudobulk samples.
# Cells are grouped by 'cell_type' and 'age', then aggregated into
# chunks of 512 cells to simulate bulk RNA-seq samples.
seu_bulk <- making_pseudobulks_from_seurat(seu,
                                           pool_by = c("cell_type", "age"),
                                           chunk_size = 512,
                                           verbose = FALSE)

# Extract the log-normalized expression matrix for prediction
lognorm_matrix <- GetAssayData(seu_bulk, assay = "RNA", layer = "data")
lognorm_matrix <- as.matrix(lognorm_matrix)

# Extract the corresponding metadata for the new pseudobulk samples
seu_bulk_meta <- seu_bulk[[c('chunk_size', 'cell_type', 'age')]]
seu_bulk_meta$age <- as.numeric(seu_bulk_meta$age)

# Apply the PASTA clock
# filter_genes = TRUE: Selects only the genes required by the PASTA model.
# rank_norm = TRUE: Applies the rank-normalization required by PASTA.
PASTA_res <- PASTA_Scores(lognorm_matrix, filter_genes = TRUE, rank_norm = TRUE)
# Prepare the data frame for plotting
plot_df <- data.frame(ActualAge = seu_bulk_meta$age,
                        Prediction = PASTA_res$PASTA)

# Create the scatter plot
ggplot(data = plot_df, aes(x = ActualAge, y = Prediction)) +
  geom_point(color = "steelblue", size = 3, alpha = 0.8) +
  geom_smooth(method = "lm", color = "black", se = FALSE) +
  ggpubr::stat_cor(method = "pearson",
                   label.x.npc = "left",
                   label.y.npc = "top",
                   hjust = 0,
                   size = 5) +
  labs(
    title = NULL,
    x = "Chronological Age",
    y = "PASTA Score"
  ) +
  theme_bw() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.title = element_text(size = 14),
    axis.text = element_text(size = 12),
    legend.position = "none"
  )
```

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



13 Tutorial Example 7: Calculate CPR and CHIP Scores

Here, we use DNAm data to calculate the CRP score and CHIP score.

```
library(OmniAgeR)
download_OmniAgeR_example("Hannum_example")
```

```
## All requested example datasets already exist.
```

```
load_OmniAgeR_example("Hannum_example")
```

```
## Dataset 'Hannum_example' loaded into environment.
```

```
CompCRP.o <- CompCRP(hannum_bmiq_m)
```

```
## [CRP] Number of represented CRP CpGs (max=1765)=1764
## [CRP] Number of represented intCRP CpGs (max=62)=62
```

```
print(lapply(CompCRP.o, head, 5))
```

```
## $crp
## [1] -0.4065040  0.3002201 -0.1465114 -0.1490234 -0.3857877
##
## $intCRP
## [1] -0.29625111  0.23775122 -0.06492197  0.03861159 -0.46205741
```

```
CompCHIP.o <- CompCHIP(hannum_bmiq_m)
```

```
## [1] "[CHIP] Number of represented AnyCHIP CpGs (max=9615)=9566"
## [1] "[CHIP] Number of represented DNMT3A CpGs (max=5990)=5960"
## [1] "[CHIP] Number of represented TET2 CpGs (max=5633)=5611"
## [1] "[CHIP] Number of represented ASXL1 CpGs (max=6078)=6025"
```

```
print(lapply(CompCHIP.o, head, 5))
```

```
## $AnyCHIP
##    GSM990532    GSM990292    GSM989979    GSM989900    GSM990054
## -0.503068906 -0.359947191  0.006808985 -0.648675130  0.353746991
##
## $DNMT3A
##  GSM990532  GSM990292  GSM989979  GSM989900  GSM990054
## -1.5684492 -0.8674698 -0.1098323 -0.3111678  1.1813856
##
## $TET2
##  GSM990532  GSM990292  GSM989979  GSM989900  GSM990054
## -0.1877774 -0.5414346  0.2673744 -0.5044092  0.1782826
##
## $ASXL1
##  GSM990532  GSM990292  GSM989979  GSM989900  GSM990054
## -0.1334423 -0.5384459  0.0903360 -0.5016612  0.1762066
```

14 Tutorial Example 8: Get CpG Probes from Various DNAm Biomarkers

```
library(OmniAgeR)
# 1. Get probes for a specific chronological aging predictor
chronological_probes <- getClockProbes(clock_names = "Horvath2013")
print(head(chronological_probes$Horvath2013))
```

```
## [1] "cg00075967" "cg00374717" "cg00864867" "cg00945507" "cg01027739"
## [6] "cg01353448"
```

```
# 2. Get probes for a specific biological aging clocks
bio_probes <- getClockProbes(clock_names = "GrimAge1")
print(head(bio_probes$GrimAge1))
```

```
## [1] "cg21789941" "cg22866430" "cg18234973" "cg26110733" "cg10636246"
## [6] "cg10578779"
```

```
# 3. Get probes for all biomaker
PCClocks_RData <- load_OmniAgeR_data(object_name = "PCClocks_data")
SystemsAge_RData <- load_OmniAgeR_data(object_name = "SystemsAge_data")
all_probes <- getClockProbes(clock_names = "all",PCClocks_RData,SystemsAge_RData)
names(all_probes)[1:10]
```

```
##  [1] "PCClocks_ALL"  "SystemsAge"    "Horvath2013"   "Hannum"
##  [5] "Lin"           "VidalBralo"    "ZhangClock"    "Horvath2018"
##  [9] "Bernabeu_cAge" "CorticalClock"
```

15 Tutorial Example 9: Celltype fraction clock based on DNAm

We apply the DNAm Cell-type Fraction (CTF) Clock to 50 blood samples (EPIC array) from the TZH cohort to predict chronological age.

```
library(OmniAgeR)
library(ggplot2)
library(patchwork)
library(ggpubr)

download_OmniAgeR_example("TZH_example_CTF")
```

```
## All requested example datasets already exist.
```

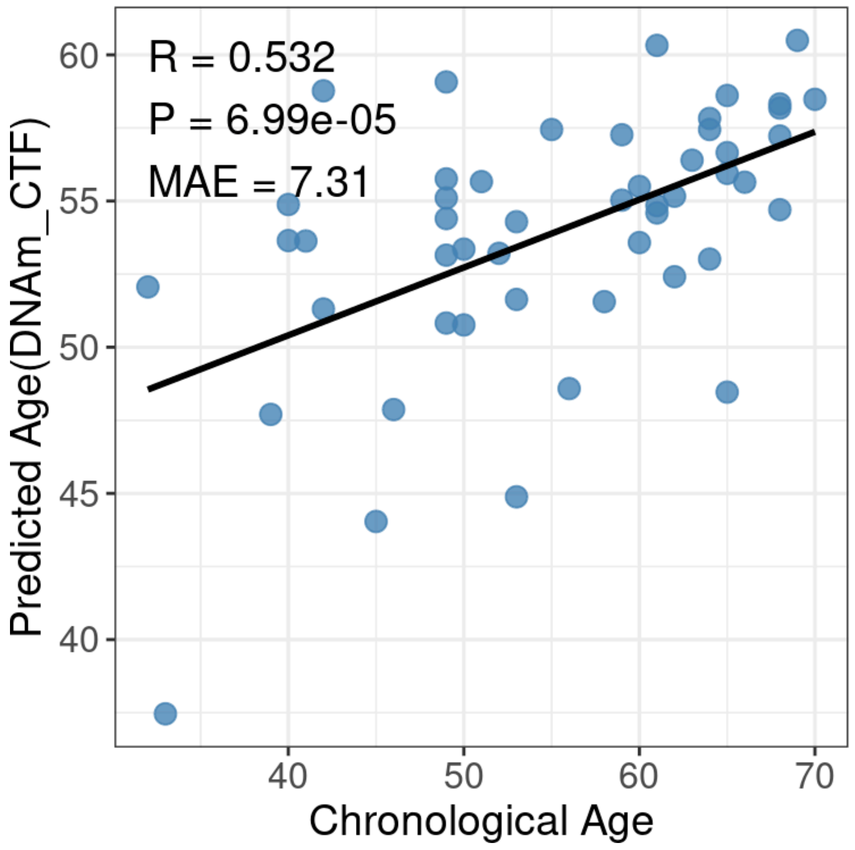
```
load_OmniAgeR_example("TZH_example_CTF")
```

```
## Dataset 'TZH_example_CTF' loaded into environment.
```

```
DNAm_CTF_Clock_o<-DNAm_CTF_Clock(CTF_m = TZH_Frac_m)
```

```
plot_df <- data.frame(ActualAge = PhenoTypes_TZH_df$Age, PredictedAge = DNAm_CTF_
cor_test_result <- cor.test(plot_df$ActualAge, plot_df$PredictedAge)
correlation <- cor_test_result$estimate
p_value <- cor_test_result$p.value
mae <- mean(abs(plot_df$PredictedAge - plot_df$ActualAge))
p_value_formatted <- ifelse(p_value < 0.001,
                          formatC(p_value, format = "e", digits = 2),
                          round(p_value, 3))
annotation_text <- paste0(
"R = ", round(correlation, 3), "\n",
"P = ", p_value_formatted, "\n",
"MAE = ", round(mae, 2)
)
ggplot(data = plot_df, aes(x = ActualAge, y = PredictedAge)) +
  geom_point(color = "steelblue", size = 3, alpha = 0.8) +
  geom_smooth(method = "lm",color = "black", se = FALSE) +
  annotate("text",
         x = min(plot_df$ActualAge, na.rm = TRUE),
         y = max(plot_df$PredictedAge, na.rm = TRUE),
         label = annotation_text,
         hjust = 0,
         vjust = 1,             size = 5) +

  labs(
    title = NULL,
    x = "Chronological Age",
    y = paste0("Predicted Age(DNAm_CTF)")
    ) +
  theme_bw() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, size = 12),
    axis.title = element_text(size = 14),
    axis.text = element_text(size = 12)
    )
```

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



```
#ggpubr::ggarrange(plotlist = gp, nrow = 1, ncol = 1)
```

```
## R version 4.5.1 (2025-06-13)
## Platform: x86_64-pc-linux-gnu
## Running under: Ubuntu 22.04.5 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.10.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.10.0  LAPACK version 3
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Asia/Shanghai
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] magrittr_2.0.4     dplyr_1.1.4        glmnet_4.1-10      Matrix_1.7-4
##  [5] Seurat_5.4.0       SeuratObject_5.3.0 sp_2.2-0           ggpubr_0.6.2
##  [9] patchwork_1.3.2    ggplot2_4.0.1      OmniAgeR_0.7.0     BiocStyle_2.38.0
##
## loaded via a namespace (and not attached):
##   [1] IRanges_2.44.0               dichromat_2.0-0.1
##   [3] locfdr_1.1-8                 goftest_1.2-3
##   [5] Biostrings_2.78.0            HDF5Array_1.38.0
##   [7] vctrs_0.6.5                  spatstat.random_3.4-3
##   [9] digest_0.6.39                png_0.1-8
##  [11] shape_1.4.6.1                proxy_0.4-29
##  [13] ggrepel_0.9.6                deldir_2.0-4
##  [15] parallelly_1.46.1            magick_2.9.0
##  [17] MASS_7.3-65                  reshape_0.8.10
##  [19] reshape2_1.4.5               httpuv_1.6.16
##  [21] foreach_1.5.2                bumphunter_1.52.0
##  [23] BiocGenerics_0.56.0          withr_3.0.2
##  [25] xfun_0.55                    survival_3.8-3
##  [27] doRNG_1.8.6.2                memoise_2.0.1
##  [29] Seqinfo_1.0.0                zoo_1.8-15
##  [31] pbapply_1.7-4                Formula_1.2-5
##  [33] KEGGREST_1.50.0              promises_1.5.0
##  [35] otel_0.2.0                   httr_1.4.7
##  [37] rstatix_0.7.3                restfulr_0.0.16
##  [39] globals_0.18.0               fitdistrplus_1.2-4
##  [41] rhdf5filters_1.22.0          stringfish_0.17.0
##  [43] rhdf5_2.54.1                 rstudioapi_0.17.1
##  [45] miniUI_0.1.2                 generics_0.1.4
##  [47] curl_7.0.0                   S4Vectors_0.48.0
##  [49] h5mread_1.2.1                polyclip_1.10-7
##  [51] randomForest_4.7-1.2         quadprog_1.5-8
##  [53] ExperimentHub_3.0.0          SparseArray_1.10.8
##  [55] xtable_1.8-4                 stringr_1.6.0
##  [57] evaluate_1.0.5               S4Arrays_1.10.1
##  [59] BiocFileCache_3.0.0          preprocessCore_1.72.0
##  [61] hms_1.1.4                    GenomicRanges_1.62.1
##  [63] bookdown_0.46                irlba_2.3.5.1
##  [65] filelock_1.0.3               ROCR_1.0-11
##  [67] qs2_0.1.6                    reticulate_1.44.1
##  [69] spatstat.data_3.1-9          lmtest_0.9-40
##  [71] readr_2.1.6                  later_1.4.5
##  [73] lattice_0.22-7               spatstat.geom_3.6-1
##  [75] future.apply_1.20.1          genefilter_1.92.0
##  [77] scattermore_1.2              XML_3.99-0.20
##  [79] cowplot_1.2.0                matrixStats_1.5.0
##  [81] RcppAnnoy_0.0.22             class_7.3-23
##  [83] pillar_1.11.1                nlme_3.1-168
##  [85] iterators_1.0.14             compiler_4.5.1
##  [87] RSpectra_0.16-2              stringi_1.8.7
##  [89] tensor_1.5.1                 SummarizedExperiment_1.40.0
##  [91] GenomicAlignments_1.46.0     plyr_1.8.9
##  [93] EpiDISH_2.26.0               crayon_1.5.3
##  [95] abind_1.4-8                  BiocIO_1.20.0
##  [97] locfit_1.5-9.12              bit_4.6.0
##  [99] codetools_0.2-20             openssl_2.3.4
## [101] bslib_0.9.0                  e1071_1.7-17
## [103] plotly_4.11.0                multtest_2.66.0
## [105] mime_0.13                    splines_4.5.1
## [107] Rcpp_1.1.1                   fastDummies_1.7.5
## [109] dbplyr_2.5.1                 sparseMatrixStats_1.22.0
## [111] knitr_1.51                   blob_1.2.4
## [113] BiocVersion_3.22.0           listenv_0.10.0
## [115] checkmate_2.3.3              DelayedMatrixStats_1.32.0
## [117] ggsignif_0.6.4               tibble_3.3.1
## [119] statmod_1.5.1                tzdb_0.5.0
## [121] pkgconfig_2.0.3              tools_4.5.1
## [123] cachem_1.1.0                 cigarillo_1.0.0
## [125] RSQLite_2.4.5                viridisLite_0.4.2
## [127] DBI_1.2.3                    fastmap_1.2.0
## [129] rmarkdown_2.30               scales_1.4.0
## [131] grid_4.5.1                   ica_1.0-3
## [133] Rsamtools_2.26.0             broom_1.0.11
## [135] AnnotationHub_4.0.0          sass_0.4.10
## [137] BiocManager_1.30.27          dotCall64_1.2
## [139] carData_3.0-5                scrime_1.3.5
## [141] RANN_2.6.2                   minfi_1.56.0
## [143] farver_2.1.2                 mgcv_1.9-4
## [145] yaml_2.3.12                  MatrixGenerics_1.22.0
## [147] rtracklayer_1.70.1           illuminaio_0.52.0
## [149] cli_3.6.5                    purrr_1.2.1
## [151] siggenes_1.84.0              stats4_4.5.1
## [153] GEOquery_2.78.0              lifecycle_1.0.5
## [155] askpass_1.2.1                uwot_0.2.4
## [157] Biobase_2.70.0               backports_1.5.0
## [159] BiocParallel_1.44.0          annotate_1.88.0
## [161] gtable_0.3.6                 rjson_0.2.23
## [163] ggridges_0.5.7               progressr_0.18.0
## [165] parallel_4.5.1               limma_3.66.0
## [167] jsonlite_2.0.0               RcppHNSW_0.6.0
## [169] bitops_1.0-9                 bit64_4.6.0-1
## [171] Rtsne_0.17                   base64_2.0.2
## [173] spatstat.utils_3.2-1         RcppParallel_5.1.11-1
## [175] jquerylib_0.1.4              spatstat.univar_3.1-5
## [177] lazyeval_0.2.2               shiny_1.12.1
## [179] htmltools_0.5.9              sctransform_0.4.3
## [181] rappdirs_0.3.3               tinytex_0.58
## [183] glue_1.8.0                   spam_2.11-3
## [185] httr2_1.2.2                  XVector_0.50.0
## [187] RCurl_1.98-1.17              mclust_6.1.2
## [189] HiBED_0.99.15                gridExtra_2.3
## [191] igraph_2.2.1                 R6_2.6.1
## [193] tidyr_1.3.2                  labeling_0.4.3
## [195] GenomicFeatures_1.62.0       cluster_2.1.8.1
## [197] rngtools_1.5.2               Rhdf5lib_1.32.0
## [199] FlowSorted.DLPFC.450k_1.46.0 beanplot_1.3.1
## [201] FlowSorted.Blood.EPIC_2.14.0 DelayedArray_0.36.0
## [203] tidyselect_1.2.1             xml2_1.5.1
## [205] car_3.1-3                    AnnotationDbi_1.72.0
## [207] future_1.68.0                KernSmooth_2.23-26
## [209] S7_0.2.1                     nor1mix_1.3-3
## [211] data.table_1.18.0            htmlwidgets_1.6.4
## [213] RColorBrewer_1.1-3           rlang_1.1.7
## [215] spatstat.sparse_3.1-0        spatstat.explore_3.6-0
## [217] rentrez_1.2.4
```