**Analysis of RAD51 and TP53**

Introduction

We separate the cell lines into two groups according to their WGD status (WGD+ and WGD–) in order to study the differential expression of genes of interest. In particular, we focus on P53, a tumor suppressor gene, and RAD51, which is involved in DNA repair. This comparison aims to determine whether the presence of a Whole Genome Duplication influences the expression of these key genes.

Methode

!/usr/bin/env python3

-\*- coding: utf-8 -\*-

"""

Created on Thu Sep 4 10:41:12 2025

@author: malos

"""

Code

import pandas as pd

df = pd.read\_csv ("/Users/malos/Desktop/hugoinfo/OmicsSignaturesProfile.csv")

Explanation

We import the pandas library, which allows us to manipulate and analyze data in tabular form. We then create a DataFrame (data table) by reading the CSV file, which enables us to open, read, and study the dataset.

Code

df\_wgd\_oui = df[df ["WGD"] == 1.0]

df\_wgd\_non = df[df ["WGD"] == 0.0]

Explanation

Here, we separate the cell lines into two groups according to the value of the **"WGD"** attribute: **df\_wgd\_oui** contains the lines where **WGD = 1.0** (presence of WGD), and **df\_wgd\_non** contains the lines where **WGD = 0.0** (absence of WGD).

Code

Print ("Lignées WGD = oui :")

Print (df\_wgd\_oui)

Print ("\nLignées WGD = non :")

Print (df\_wgd\_non)

Explanation

This script successively displays two subsets of the original table: first, the cell lines where the **WGD column = 1.0**, corresponding to lines with a Whole Genome Duplication (WGD), and then the cell lines where **WGD = 0.0**, corresponding to lines without a Whole Genome Duplication. For better readability, a title is displayed before each subset, and a line break separates the two outputs.

Code

df\_wgd\_oui. to\_csv ("cell\_lines\_WGD\_oui.csv", index=False)

df\_wgd\_non. to\_csv ("cell\_lines\_WGD\_non.csv", index=False)

Explanation

This script saves the two data subsets into separate CSV files: **cell\_lines\_WGD\_oui.csv** contains the cell lines for which **WGD = 1.0** (presence of a Whole Genome Duplication), and **cell\_lines\_WGD\_non.csv** contains the cell lines for which **WGD = 0.0** (absence of a Whole Genome Duplication). The parameter index=False ensures that the DataFrame row index is not included in the file, so that only the relevant columns are saved.

Code

**Print ("Fichiers créés :")**

**Print (" cell\_lines\_WGD\_oui.csv")**

**Print (" cell\_lines\_WGD\_non.csv”)**

**Explanation**

This code displays a confirmation message on the screen, indicating that the two CSV files have been generated: **cell\_lines\_WGD\_oui.csv** (containing the lines with WGD) and **cell\_lines\_WGD\_non.csv** (containing the lines without WGD), thereby informing the user that the data export was successful.

Code

import pandas as pd

import matplotlib.pyplot as plt

Explanation

Here, this code imports the libraries **pandas** (to manipulate and analyze tabular data) and **matplotlib.pyplot** (to create graphs and visualize the results).

Code

df\_final = pd.read\_csv("/Users/malos/Desktop/hugoinfo/merged\_TP53\_RAD51\_WGD\_norm.csv")

print (df\_final.columns.tolist())

Explanation

Here, this code reads the CSV file merged\_TP53\_RAD51\_WGD\_norm.csv into a DataFrame named **df\_final**, and then displays the list of column names in this table to verify the imported data.

**Code**

**df\_oui = df\_final[df\_final ["WGD"] == "oui"]**

**df\_non = df\_final[df\_final ["WGD"] == "non"]**

Explanation

Here, this code separates the data into two subsets: **df\_oui** contains the cell lines where the WGD column is equal to "oui" (presence of WGD), and **df\_non** contains the cell lines where the WGD column is equal to "non" (absence of WGD).

Code

plt.figure (figsize=(6, 5))

plt.scatter(df\_oui["p53"], df\_oui["RAD51"], alpha=0.6, color="blue")

plt.xlabel("Expression RAD51")

plt.ylabel("TP53 gene effect")

plt.title("RAD51 expression vs TP53 knockout (CRISPR) WGD +")

plt.grid (True, linestyle="--", alpha=0.5)

Explanation

Here, this code creates a scatter plot representing the **WGD+** cell lines: the x-axis corresponds to **RAD51** expression, the y-axis corresponds to the **TP53** gene effect (CRISPR knockout), and each blue point represents a cell line. The plot is formatted with a title, axis labels, and a grid to facilitate readability.

Code

plt.figure (figsize=(6, 5))

plt.scatter(df\_non["p53"], df\_non["RAD51"], alpha=0.6, color="red")

plt.xlabel("Expression RAD51")

plt.ylabel("TP53 gene effect")

plt.title("RAD51 expression vs TP53 knockout (CRISPR) WGD -")

plt.grid (True, linestyle="--", alpha=0.5)

Explanation

Here, this code creates a scatter plot representing the **WGD–** cell lines: the x-axis corresponds to **RAD51** expression, the y-axis corresponds to the **TP53** gene effect (CRISPR knockout), and each red point represents a cell line. The plot is formatted with a title, axis labels, and a dashed grid to improve readability.

Result

Une image contenant texte, capture d’écran, diagramme, Tracé

Le contenu généré par l’IA peut être incorrect.

**Figure 1:** Relationship between RAD51 expression and TP53 knockout (CRISPR) effect in WGD+ cell lines.

This scatter plot shows the relationship between **RAD51** expression (x-axis) and the effect of **TP53** knockout by CRISPR (y-axis) in cell lines presenting a Whole Genome Duplication (WGD+). Each blue point represents a cell line.

Une image contenant texte, capture d’écran, diagramme, Tracé

Le contenu généré par l’IA peut être incorrect.

**Figure 2:** Relationship between RAD51 expression and TP53 knockout (CRISPR) effect in WGD– cell lines.

This scatter plot shows the relationship between **RAD51** expression (x-axis) and the effect of **TP53** knockout by CRISPR (y-axis) in cell lines without a Whole Genome Duplication (WGD–). Each red point represents a cell line.

Code python

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

"""

Created on Thu Sep 4 10:41:12 2025

@author: malos

"""

import pandas as pd

df = pd.read\_csv("/Users/malos/Desktop/hugoinfo/OmicsSignaturesProfile.csv")

df\_wgd\_oui = df[df["WGD"] == 1.0]

df\_wgd\_non = df[df["WGD"] == 0.0]

print("Lign√©es WGD = oui :")

print(df\_wgd\_oui)

print("\nLign√©es WGD = non :")

print(df\_wgd\_non)

df\_wgd\_oui.to\_csv("cell\_lines\_WGD\_oui.csv", index=False)

df\_wgd\_non.to\_csv("cell\_lines\_WGD\_non.csv", index=False)

print("Fichiers cr√©√©s :")

print(" cell\_lines\_WGD\_oui.csv")

print(" cell\_lines\_WGD\_non.csv")

import pandas as pd

import matplotlib.pyplot as plt

df\_final = pd.read\_csv("/Users/malos/Desktop/hugoinfo/merged\_TP53\_RAD51\_WGD\_norm.csv")

print(df\_final.columns.tolist())

df\_oui = df\_final[df\_final["WGD"] == "oui"]

df\_non = df\_final[df\_final["WGD"] == "non"]

plt.figure(figsize=(6, 5))

plt.scatter(df\_oui["p53"], df\_oui["RAD51"], alpha=0.6, color="blue")

plt.xlabel("Expression RAD51")

plt.ylabel("TP53 gene effect")

plt.title("RAD51 expression vs TP53 knockout (CRISPR) WGD +")

plt.grid(True, linestyle="--", alpha=0.5)

plt.figure(figsize=(6, 5))

plt.scatter(df\_non["p53"], df\_non["RAD51"], alpha=0.6, color="red")

plt.xlabel("Expression RAD51")

plt.ylabel("TP53 gene effect")

plt.title("RAD51 expression vs TP53 knockout (CRISPR) WGD -")

plt.grid(True, linestyle="--", alpha=0.5)

Copier coller dans python