

Phi-CRISPR: Arquitectura Teórica Preliminar

LASU

9 de diciembre de 2025

Resumen

El presente informe desarrolla el marco conceptual, técnico y bioético del sistema Phi-CRISPR, una arquitectura teórica avanzada concebida como evolución de las tecnologías CRISPR convencionales. Su diseño se orienta a maximizar la precisión, la estabilidad funcional y el control espacio-temporal de la edición genómica, integrando principios de regulación epigenética y modulación génica programable. El documento mantiene un enfoque estrictamente teórico, excluyendo deliberadamente protocolos experimentales o instrucciones técnicas que permitan su reproducción operativa, en consonancia con criterios de responsabilidad científica y normativas bioéticas internacionales.

1 Introducción General al Sistema Phi-CRISPR

Phi-CRISPR se plantea como una arquitectura conceptual destinada a ofrecer un control refinado sobre la regulación génica, integrando mecanismos epigenéticos, señales contextuales y moduladores programables. Su objetivo central es lograr intervenciones de alta precisión sin recurrir a mecanismos operativos no regulados. Se presenta como un marco teórico para exploración científica responsable.

2 Módulo Terapéutico Conceptual: Eje p53–BRCA1

Este módulo describe una propuesta conceptual destinada a comprender, modelar y justificar estrategias teóricas para potenciar las funciones de TP53 y BRCA1 sin detallar procedimientos experimentales.

2.1 Objetivo del módulo

2.1.1. Ficha conceptual: Bifunción p53–BRCA1

El objetivo es diseñar una entidad farmacológica, ya sea una única molécula bifuncional o una combinación controlada, que cumpla dos funciones complementarias. La primera consiste en estabilizar y/o reactivar la actividad transcripcional de p53 (TP53) en células que conserven p53 funcional o variantes susceptibles de reactivación. La segunda es aumentar la expresión funcional o la actividad operativa de BRCA1 mediante la remoción de silenciamiento epigenético del promotor o la estabilización de complejos cofactores que favorezcan su transcripción o estabilidad proteica.

La motivación de este diseño radica en la complementariedad biológica entre ambas proteínas. Mientras p53 actúa como centinela del daño en el ADN y regulador clave de apoptosis y arresto del ciclo ce-

lular, BRCA1 desempeña un papel central en la reparación por recombinación homóloga. Potenciar simultáneamente estas funciones incrementaría la capacidad de reparación y reduciría la supervivencia de clones celulares con inestabilidad genómica.

- Estabilice y/o reactive la actividad transcripcional de p53 (TP53).
- Aumente la expresión o función operativa de BRCA1 mediante regulación epigenética o estabilización proteica.

Motivación: p53 actúa como centinela de daño en ADN; BRCA1 es clave en la recombinación homóloga. Su potenciación conjunta favorece estabilidad genómica.

2.2 Estrategias moleculares conceptuales

A. Inhibidor competitivo de p53–MDM2 (tipo Nutlin). B. Reactivador conceptual de p53 mutante (APR-246 / eprenetapopt). C. “Molecular glue” para estabilizar interacciones que favorezcan BRCA1. D. Modulador epigenético dirigido para reducir silenciamiento del promotor BRCA1. E. Diseño híbrido: combinación conceptual de elementos de las estrategias anteriores.

2.3 Propiedades físico-químicas objetivo

Rangos guía para diseño in silico:

- MW <550 Da (monofuncional).
- LogP entre 0 y 4.
- PSA <140 Å²; ideal <110 Å² para acceso nuclear.
- Estado de carga neutro o ligeramente básico.
- Cumplimiento flexible de criterios de drug-likeness.

3 Diseño estructural conceptual

Tres fragmentos funcionales componen el esquema bifuncional propuesto. **Dominio A (ancla PPI):** fragmento mimético del motivo de p53 capaz de ocupar el bolsillo de MDM2 o interactuar con su superficie, reduciendo la degradación de p53. **Enlace/S-pacer:** segmento flexible o semirrígido que permita que la otra mitad alcance su blanco sin obstaculizar la unión del Dominio A, con longitud y rigidez optimizadas mediante modelado conformacional. **Dominio B (efector BRCA1):** dos alternativas conceptuales. B1: superficie farmacófora que actúe como molecular glue reclutando un coactivador transcripcional sobre regiones reguladoras de BRCA1. B2: módulo capaz de favorecer la activación local de cromatina mediante interacción con complejos que disminuyan metilación o aumenten acetilación sobre el promotor de BRCA1.

- B1: superficie tipo molecular glue.
- B2: módulo conceptual de interacción epigenética local.

3.1 Evaluación computacional (conceptual)

Las actividades in silico recomendadas incluyen:

- Docking a MDM2 o variantes de p53.
- Modelado de ternary complex para glues.
- Dinámica molecular y estimación de energía libre.
- ADMET in silico.
- QSAR y modelos ML para priorización.

3.2 Biomarcadores conceptuales

- Aumento de p21 (CDKN1A) como señal de p53 funcional.
- Incremento de BRCA1 a nivel mRNA y proteína.
- Mejora de foci de RAD51.
- Reducción de H2AX en dinámica temporal.

3.3 Riesgos y puntos críticos

- Activación indiscriminada de p53/BRCA1 en tejidos sanos.
- Ineficacia en tumores con deleciones estructurales.
- Posibles off-targets en glues.
- Limitaciones farmacocinéticas por tamaño molecular.

3.4 Alternativas conceptuales

- Terapia combinada conceptual: reactivador p53 + modulador epigenético BRCA1.
- Enfoque de protein rescue para BRCA1.
- Modulación génica no proteica (ARN/epigenética).

4 Anexos

4.1 SDRP: Sistema de Detección y Reconocimiento de Patologías

El SDRP es un módulo conceptual de análisis de imágenes para evaluar cambios morfológicos mediante algoritmos de reconocimiento visual.

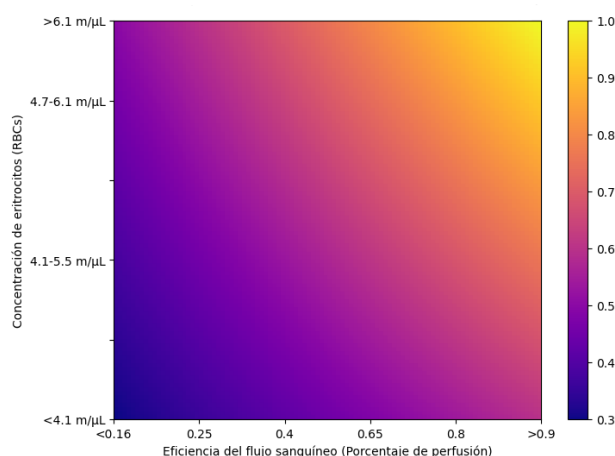


Figura 1: Aproximación genérica por SDRP durante etapa 3 en función de RBCs.

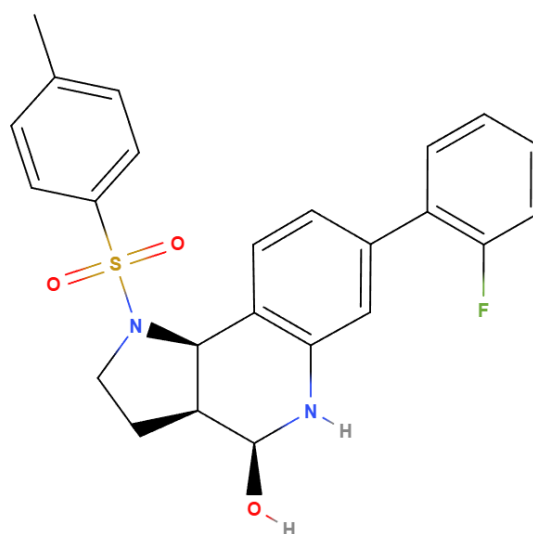


Figura 2: Inhibidor BRD0539 (referencia ilustrativa).

Listing 1: Boceto estructural de SDRP (referencial)

```

1  pip install opencv-python numpy
   matplotlib
2  import cv2
3  import numpy as np
4  try:
5      image_path1 = 'image1.jpg'
6      image_path2 = 'image2.jpg'
7      img1_original = cv2.imread(
8          image_path1)
9      img2_original = cv2.imread(
10         image_path2)
11     if img1_original is None or
12        img2_original is None:
13         print("Creating dummy images for
14             demonstration as actual
15             images were not found.")
16         dummy_img_shape = (400, 600, 3)
17         img1_original = np.zeros(
18             dummy_img_shape, dtype=np.
19             uint8) + 50
20         img2_original = np.zeros(
21             dummy_img_shape, dtype=np.
22             uint8) + 100
23         img2_original[100:150, 100:200]
24         = [0, 0, 255]
25         cv2.imwrite(image_path1,
26             img1_original)
27         cv2.imwrite(image_path2,
28             img2_original)
29         print(f"Dummy images '{
30             image_path1}' and '{
31             image_path2}' created.")
32         img1_original = cv2.imread(
33             image_path1)
34         img2_original = cv2.imread(
35             image_path2)
36     except Exception as e:
37         print(f"An error occurred while
38             loading/creating images: {e}")
39         print("Creating default dummy images
40             .")
41         dummy_img_shape = (400, 600, 3)
42         img1_original = np.zeros(
43             dummy_img_shape, dtype=np.uint8)
44         + 50
45         img2_original = np.zeros(
46             dummy_img_shape, dtype=np.uint8)
47         + 100
48         img2_original[100:150, 100:200] =
49         [0, 0, 255]
50     if img1_original is None:
51         print(f"Error: Could not load image
52             from '{image_path1}'. Please
53             check the path.")
54     if img2_original is None:
55         print(f"Error: Could not load image
56             from '{image_path2}'. Please
57             check the path.")
58     if img1_original is not None and
59        img2_original is not None:
60         print("Both images loaded
61             successfully.")
62         print(f"Image 1 original shape: {
63             img1_original.shape}")
64         print(f"Image 2 original shape: {
65             img2_original.shape}")
66         img1_gray = cv2.cvtColor(
67             img1_original, cv2.COLOR_BGR2GRAY
68         )
69         img2_gray = cv2.cvtColor(
70             img2_original, cv2.COLOR_BGR2GRAY
71         )
72         print("Images converted to grayscale
73             .")
74         if img1_gray.shape != img2_gray.
75             shape:
76             print("Image dimensions differ.
77                 Resizing...")
78             target_height, target_width =
79                 img1_gray.shape
80             img2_gray_resized = cv2.resize(
81                 img2_gray, (target_width,
82                     target_height), interpolation
83                     =cv2.INTER_AREA)
84             img1_processed = img1_gray
85             img2_processed =
86                 img2_gray_resized
87             print(f"Image 2 resized from {
88                 img2_gray.shape} to {
89                 img2_processed.shape}.")
90         else:
91             img1_processed = img1_gray
92             img2_processed = img2_gray
93             print("Images already have
94                 compatible dimensions.")
95             print("Preprocessed images stored in
96                 'img1_processed' and '
97                 img2_processed'.")
98     else:
99         print("Cannot proceed with image
100             preprocessing due to loading
101             errors.")
102     from skimage.metrics import
103         structural_similarity as ssim
104     (score, diff) = ssim(img1_processed,
105         img2_processed, full=True)
106     print(f"SSIM Score: {score:.4f}")
107     diff = (diff * 255).astype("uint8")
108     diff_normalized_uint8 = diff
109     print("Difference image calculated and
110         normalized to uint8 as '
111         diff_normalized_uint8'.")
112     import matplotlib.pyplot as plt
113     fig, axes = plt.subplots(3, 2, figsize
114         =(15, 15))
115     axes[0, 0].imshow(cv2.cvtColor(
116         img1_original, cv2.COLOR_BGR2RGB))
117     axes[0, 0].set_title('Original Image 1')
118     axes[0, 0].axis('off')
119     axes[0, 1].imshow(cv2.cvtColor(
120         img2_original, cv2.COLOR_BGR2RGB))
121     axes[0, 1].set_title('Original Image 2')
122     axes[0, 1].axis('off')
123     axes[1, 0].imshow(img1_processed, cmap='
124         gray')
125     axes[1, 0].set_title('Processed Image 1
126         (Grayscale)')
127     axes[1, 0].axis('off')
128     axes[1, 1].imshow(img2_processed, cmap='
129         gray')
130     axes[1, 1].set_title('Processed Image 2

```

```

    (Grayscale)')
71 axes[1, 1].axis('off')
72 axes[2, 0].imshow(diff_normalized_uint8,
    cmap='gray')
73 axes[2, 0].set_title('Difference Image (
    SSIM)')
74 axes[2, 0].axis('off')
75 thresh = cv2.threshold(
    diff_normalized_uint8, 0, 255, cv2.
    THRESH_BINARY_INV | cv2.THRESH_OTSU)
    [1]
76 thresh_copy = thresh.copy()
77 contours, _ = cv2.findContours(
    thresh_copy, cv2.RETR_EXTERNAL, cv2.
    CHAIN_APPROX_SIMPLE)
78 img1_diff_highlighted = img1_original.
    copy()
79 for c in contours:
80     if cv2.contourArea(c) > 100:
81         (x, y, w, h) = cv2.boundingRect(
            c)
82         cv2.rectangle(

```

```

    img1_diff_highlighted, (x, y)
    , (x + w, y + h), (0, 0, 255)
    , 2)
83 axes[2, 1].imshow(cv2.cvtColor(
    img1_diff_highlighted, cv2.
    COLOR_BGR2RGB))
84 axes[2, 1].set_title('Image 1 with
    Highlighted Differences')
85 axes[2, 1].axis('off')
86 plt.tight_layout()
87 plt.show()

```

5 Referencias y bibliografía

1. Liu D. Targeting p53–MDM2 interaction by small-molecule inhibitors. J Hematol Oncol. 2008.
2. [Artículo 91] Targeting p53–MDM2 interaction by small-molecule inhibitors: learning from MDM2 inhibitors in clinical trials. J Hematol Oncol. 2022;15:91.