Supplementary information

An Investigation on Biophysical parameters and Role of Magnesium concentration on spCas9 interaction with Target and Off-Target sequence

Deependra Kumar Ban[1#], Kshama Parate[1#], Deepta Bharadwaj[1], Austin Wong[1], Lorelai Schoch[1], Kenneth Visk[1], Kiana Aran[1,2*]

[1] Shu Chien-Gene Lay Department of Bioengineering, Jacobs School of Engineering Department of Bioengineering, University of California, San Diego, CA, USA 92093

[2]Department of Medicine, Division of Geriatrics, Gerontology & Palliative Care, School of Medicine, University of California, San Diego, CA, USA 92093

#Equal first author

*Corresponding author email: karan@ucsd.edu

Table S1: Methods for off-target detection, their advantages and disadvantages

Method	**Description**	**Advantages**	**Disadvantages**
qEva-CRISPR[1]	Detection of multiple mutations and deletions	Can access difficult to reach genomic regions	Cannot detect off-target sites in whole genome or accidental off-target sites
GUIDE-seq[2]	Global detection of double-stranded breaks by any nuclease	Genome wide identification of off-target effects	Capability limited by sequencing depth
SMRT Sequencing[3]	Quantification of genome-editing outcome at any loci with various engineered nuclease platforms	Long reads with capability to recognize long inserts at high sensitivity	High error rate, low throughput and high cost[4]
PEACE-seq[5]	Enriching local sequences of edited sites to identify off-target sequence	Demonstrated in vivo application	Might overlook cleavage with incomplete insertions
DISCOVER-seq, ChIP-seq[6,7]	Chromatin immunoprecipitation enriches DNA fragments to which specific protein is	High resolution of single base pair, greater coverage and large dynamic range, applicable in ex-vivo	DISCOVER-seq requires DNA-PKcs inhibitor for activity. ChIP-seq captures snapshots of MRE11

	bound	and in-vivo systems	binding, may not exhibit off-target sites, false positives
Tracking of Indels by Decomposition (TIDE)[8]	Sanger Sequencing of amplified DNA, indels analysis using TIDE software	Low cost with 2 PCR steps, detect indels with sensitivity of 1-2% of target in a pool of cells	Low throughput, output depends on purity of PCR products
DeepCRISPR[9]	Deep learning computational platform for on-target and off-target site prediction and optimizes gRNA design	Algorithm improvement with training data	Unsuited for base editors and prime editors

Table S2: list of dsDNA sequences: single nucleotide mismatch healthy sequence, HU, sickle cell specific target, SCDU; and guide RNA (gRNA) sequence for SCDU.

Name	Sequence	References
Healthy sequence (WD)	ThioMC6-D-ACC TCA AAC AGA CAC CAT GGT GCA TCT GAC TCC TGA GGA GAA GTC TGC CGT TAC TGC CCT GTG-3' and	[10]
Sickel cell specific sequence (SCD)	ThioMC6-D-ACC TCA AAC AGA CAC CAT GGT GCA TCT GAC TCC TGT GGA GAA GTC TGC CGT TAC TGC CCT GTG-3'	[10]
gRNA sequence	5′-GUAACGGCAGACUUCUCCAC-3'	[10]
Random DNA	5'-TAGGTGATTTTGGTCTAGCTACAGTGGAATCTCGATGGAGTGGGTCCCAT-3'	
Random gRNA	5'-CUUACCUCCAGAUAUAUUGA-3'	

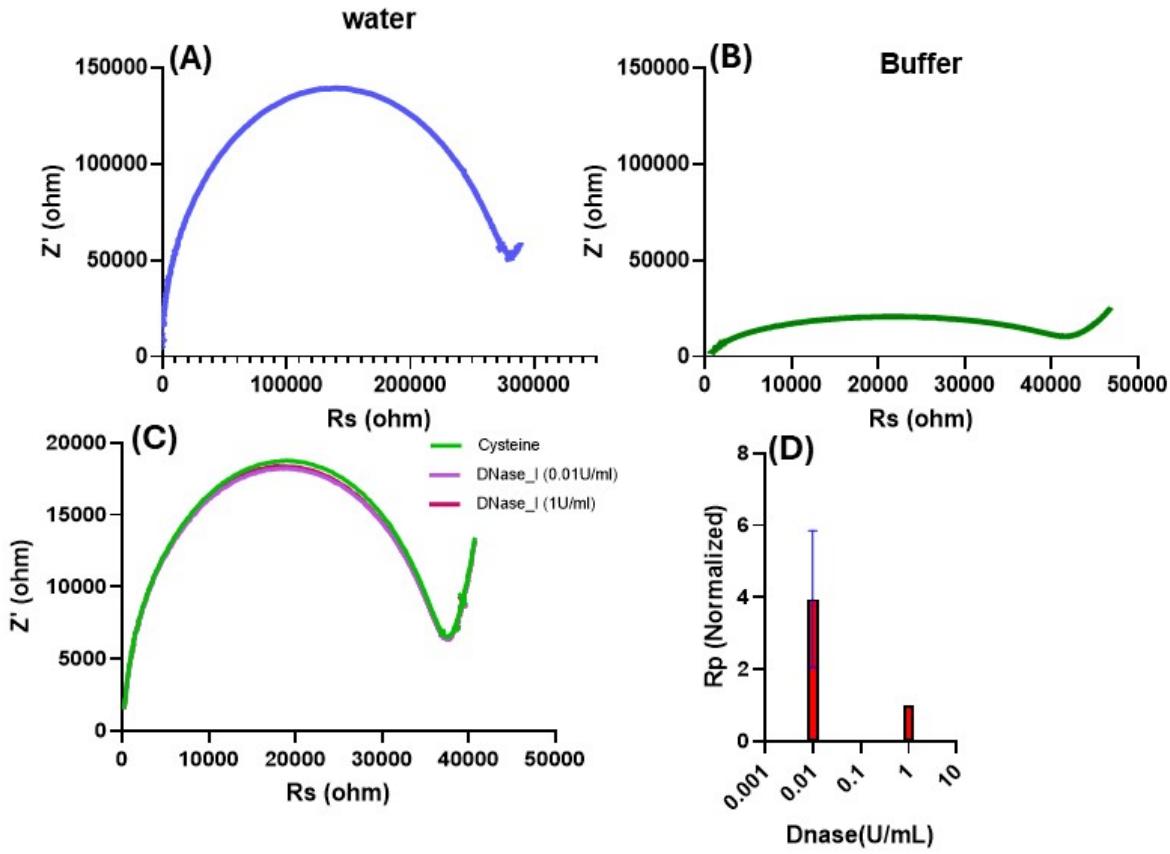

Figure S1: (A) Nyquist plot for ion free water showing large impedance and interfacial charge transfer resistance (R_p). (B) exposure of 0.01xTris-HCl causes reduced R_p to seven-fold. (C)-(D) showing Dnase_I interaction with dsDNA conjugated working electrode. The results showed approximately 2fold decrease in Rp value when exposed to 0.01U/mL Dnase to 1 U/mL.

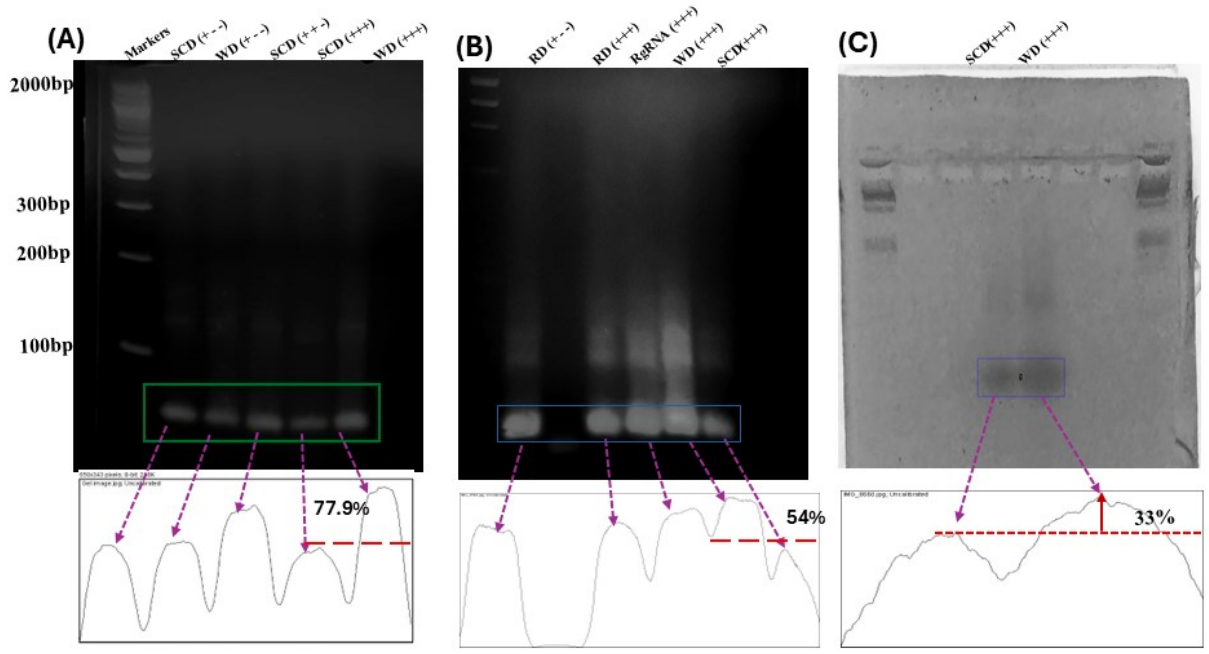

Figure S2: (A) and (C) Gel shift/cleavage assay. Perfectly matched SCD, seed-region SNP containing WD, and random (RD) dsDNA substrates were incubated with 100 nM spCas9 or spCas9-gRNA for 30 min. Where indicated, spCas9 was complexed with a random gRNA (RgRNA) and incubated with SCD (15 µL of 10 µM). (B) we have loaded 40 µl (10 uM) of samples. Reaction products were resolved on a 4% agarose gel, stained with ethidium bromide, and visualized under UV illumination. The spCas9-gRNA + SCD lane shows a pronounced reduction in the full-length DNA band compared with controls, including WD only (+ − −), WD+spCas9-gRNA (+++), SCD+*sp*Cas9(++ −), RD+*sp*Cas9-gRNA (+++), and SCD+*sp*Cas9-RgRNA (+++), confirming selective cleavage of the perfectly matched target. Quantitative analysis revealed a 34-77% reduction in SCD band intensity upon incubation with spCas9-gRNA relative to WD, RD, or RgRNA controls, demonstrating the high specificity of the selected gRNA.

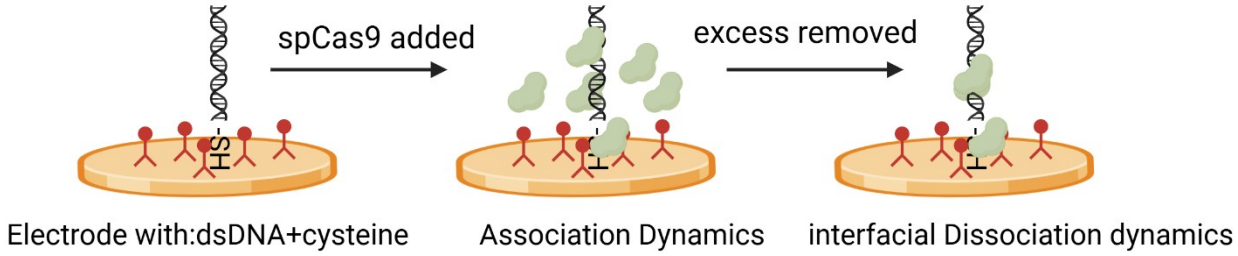

Figure S3: Stepwise kinetic analysis of localized spCas9–dsDNA interactions measured by nfEIS. Gold electrodes functionalized with dsDNA and cysteine were incubated with spCas9 for 30 min to monitor the association phase. Following incubation, excess spCas9 was removed, and

a fresh reaction buffer containing 5 mM $MgCl_2$ was introduced to initiate continuous monitoring of dsDNA dissociation dynamics over a 1 h period.

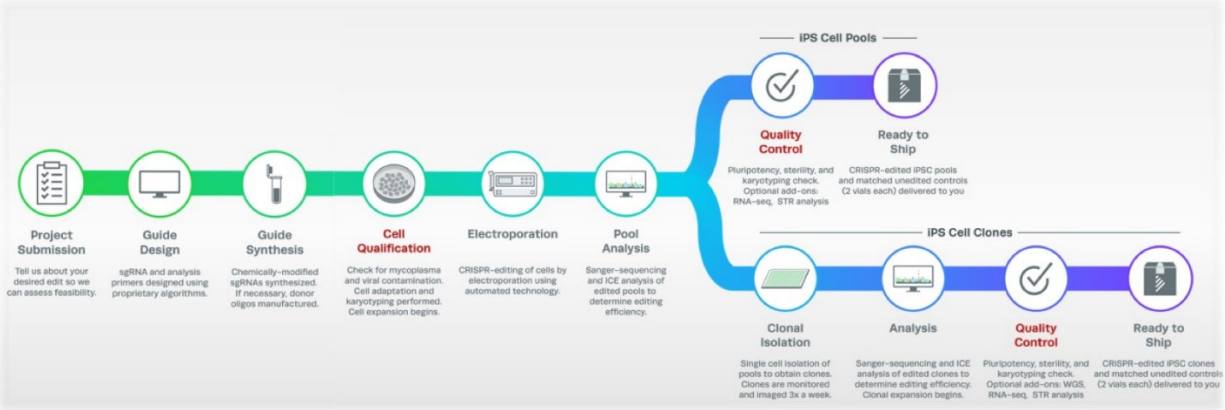

Figure S4. Proposed integration of nfEIS-based prescreening within a standardized CRISPR genome-editing workflow. Schematic representation of a typical genome-editing pipeline, including guide design, guide synthesis, cell qualification, electroporation, pool analysis, clonal isolation, and quality control. We propose incorporation of nfEIS-based, label-free on-chip evaluation immediately after guide synthesis and prior to cellular delivery. In this configuration, Cas9-gRNA ribonucleoprotein complexes are evaluated for binding affinity, association/dissociation kinetics, cleavage dynamics, and residence time using immobilized DNA targets. This prescreening step enables rapid ranking and prioritization of gRNA candidates before advancing selected guides to electroporation and downstream cellular validation.

Python code used to calculate Rp values:

```
import os

import pandas as pd

import numpy as np

import xlsxwriter

# ================================================================

# Purpose:

#   Script to process processed chip Excel files and

#   generate a single combined Excel file:

#   "Big Excel normalized Rp_d Subtracted.xlsx"

#

#   Only extracts and combines "normalized Rp_d" data for
```

```
#   Cas_complex, MgCl2 = 1 concentration chips.
# ====================================================================
def main():
    # --- 1. Setup base directory ---
    try:
        base_dir = os.path.dirname(os.path.abspath(__file__))
    except NameError:
        base_dir = os.getcwd()
    # Folder where processed chip Excel files are stored
    processed_chips_folder_path = os.path.join(base_dir, "Processed Chips for PCAI1ia")
    os.makedirs(processed_chips_folder_path, exist_ok=True)
    folder_path = processed_chips_folder_path
    print(f"Using folder: {folder_path}")
    # --- 2. Define valid worksheets to read from ---
    worksheet_names = [
        "0pM_asso", "100pM_asso", "1nM_asso", "10nM_asso", "100nM_asso",
        "0pM_disso", "100pM_disso", "1nM_disso", "10nM_disso", "100nM_disso"
    ]
    # --- 3. Columns to extract (only what's needed) ---
    headers = ['time(mins)', 'normalized Rp_d']
    # --- 4. Collect all valid chip Excel files ---
    excel_files = [f for f in os.listdir(folder_path) if f.endswith('.xlsx') and f.startswith('Chip')]
    if not excel_files:
        print("No valid Excel files found in folder.")
        return
    # --- 5. Read and combine data from all chips ---
    all_data = []
```

```
for file_name in excel_files:
    file_path = os.path.join(folder_path, file_name)
    chip_name = os.path.splitext(file_name)[0]  # e.g., "Chip 35"
    xl = pd.ExcelFile(file_path)
    for sheet in xl.sheet_names:
        if sheet not in worksheet_names:
            continue
        # Read only the relevant columns if they exist
        df = xl.parse(sheet, usecols=lambda col: col in headers)
        if 'time(mins)' not in df.columns or 'normalized Rp_d' not in df.columns:
            continue
        df['chip'] = chip_name
        df['sheet'] = sheet
        all_data.append(df)
if not all_data:
    print("No valid 'normalized Rp_d' data found.")
    return
df_all = pd.concat(all_data, ignore_index=True)
# --- 6. Subset only Cas_complex chips at MgCl2 = 1 ---
# (Assumes a separate file df_sub exists or can be defined here)
# Since no chip info dictionary is used, we'll filter later if needed.
df_sub = df_all.copy()  # Keep same variable name for consistency
# --- 7. Filter: only concentration 1, Cas_complex type ---
# Adjust if your dataframe already contains Type/Concentration columns
# If not, skip or add filtering after merging metadata.
df_mgcl1_com = df_sub  # ← Placeholder: keep entire df_sub if no metadata present
# --- 8. Pivot to create final summary structure ---
```

```
value_col = "normalized Rp_d"
print(f"\nProcessing: {value_col} (Subtracted)\n")
pivot_df = df_mgcl1_com.pivot_table(
    index='time(mins)',
    columns=['sheet', 'chip'],
    values=value_col,
    aggfunc='first'
).reset_index()
# --- 9. Save final Big Excel file ---
output_file = os.path.join(folder_path, f"Big Excel {value_col} Subtracted1.xlsx")
with pd.ExcelWriter(output_file, engine='xlsxwriter',
                    engine_kwargs={'options': {'nan_inf_to_errors': True}}) as writer:
    workbook = writer.book
    worksheet = workbook.add_worksheet("Summary")
    # Sort columns for consistent ordering
    existing_cols = pivot_df.columns.drop('time(mins)')
    existing_cols = sorted(existing_cols, key=lambda x: (
        worksheet_names.index(x[0]) if x[0] in worksheet_names else 999,
        x[1]
    ))
    pivot_df_ordered = pivot_df[['time(mins)']].join(pivot_df[existing_cols])
    # --- Write headers ---
    header_format = workbook.add_format({'bold': True, 'border': 1,
                                         'align': 'center', 'valign': 'vcenter'})
    cols = pivot_df_ordered.columns
    worksheet.write(0, 0, "time(mins)")
    for j, col in enumerate(cols[1:], start=1):
```

```
            worksheet.write(0, j, col[0])  # sheet name
        worksheet.write(1, 0, "")
        for j, col in enumerate(cols[1:], start=1):
            worksheet.write(1, j, col[1])  # chip name
        # --- Write data rows ---
        for i, row_data in enumerate(pivot_df_ordered.values.tolist()):
            row_data = ['' if pd.isna(x) or np.isinf(x) else x for x in row_data]
            worksheet.write_row(i + 2, 0, row_data)
    print(f" Excel saved to: {output_file}")
    print("Done.")
if __name__ == "__main__":
    main()
```

References:

1 M. Dabrowska, K. Czubak, W. Juzwa, W. J. Krzyzosiak, M. Olejniczak and P. Kozlowski, *Nucleic Acids Res.*, 2018, **46**, e101–e101.

2 S. Q. Tsai, Z. Zheng, N. T. Nguyen, M. Liebers, V. V. Topkar, V. Thapar, N. Wyvekens, C. Khayter, A. J. Iafrate, L. P. Le, M. J. Aryee and J. K. Joung, *Nat. Biotechnol.*, 2015, **33**, 187–197.

3 A. Hendel, E. J. Kildebeck, E. J. Fine, J. T. Clark, N. Punjya, V. Sebastiano, G. Bao and M. H. Porteus, *Cell Rep.*, 2014, **7**, 293–305.

4 S. Pavlovic, K. Klaassen, B. Stankovic, M. Stojiljkovic and B. Zukic, in *Microbiomics*, Elsevier, 2020, pp. 175–200.

5 Z. Yu, Z. Lu, J. Li, Y. Wang, P. Wu, Y. Li, Y. Zhou, B. Li, H. Zhang, Y. Liu and L. Ma, *Nat. Commun.*, 2022, **13**, 7545.

6 R. S. Zou, Y. Liu, O. E. R. Gaido, M. F. Konig, B. J. Mog, L. L. Shen, F. Aviles-Vazquez, A. Marin-Gonzalez and T. Ha, *Nat. Methods*, 2023, **20**, 706–713.

7 P. J. Park, *Nat. Rev. Genet.*, 2009, **10**, 669–680.

8 E. K. Brinkman, T. Chen, M. Amendola and B. van Steensel, *Nucleic Acids Res.*, 2014, **42**, e168–e168.

9 G. Chuai, H. Ma, J. Yan, M. Chen, N. Hong, D. Xue, C. Zhou, C. Zhu, K. Chen, B. Duan, F. Gu, S. Qu, D. Huang, J. Wei and Q. Liu, *Genome Biol.*, 2018, **19**, 80.

10 S. Balderston, J. J. Taulbee, E. Celaya, K. Fung, A. Jiao, K. Smith, R. Hajian, G. Gasiunas, S. Kutanovas, D. Kim, J. Parkinson, K. Dickerson, J.-J. Ripoll, R. Peytavi, H.-W. Lu, F. Barron, B. R. Goldsmith, P. G. Collins, I. M. Conboy, V. Siksnys and K. Aran, *Nat. Biomed. Eng.*, 2021, **5**, 713–725.