

Correcting for Survival Bias in the Systems Biology of Social Isolation

Ofir Razon and Matan Kichler

Systems Biology of Aging – Final Project – February 2026

Question

The "Saturated Removal" (SR) model describes aging as a stochastic process governed by the balance between damage production (η) and removal capacity (β). In **Lecture 12: Friends, Stress and Wisdom**, we learned that social isolation is a physiological stressor known to elevate cortisol and suppress immune function, which in turn accelerates the accumulation of damage (X), leading to earlier mortality.

Using a massive epidemiological dataset (NHANES 1999–2018, $N > 27,000$), we investigate the impact of social isolation (living alone) on mortality. Crucially, we address common methodological pitfalls including Left Truncation Bias and hidden demographic drivers.

We ask the following multi-section question:

1. **The Naive Analysis:** When analyzing the pooled population (ignoring entry age), does social isolation appear to decrease survival, and does this aggregation obscure specific sex-dependent dynamics?
2. **The "Immortal" Bias:** Isolating the female cohort, we observe that those living alone are significantly older at baseline than those living with others. How does this introduce a "Left Truncation" bias that might underestimate the true mortality risk?
3. **The Corrected Dynamic & Sex Vulnerability:** How can we mathematically correct the survival function $S(t)$ to account for delayed entry? Once corrected, does the "protective" effect in females vanish, and is there a significant difference in the severity of the isolation penalty between sexes?
4. **Age-Dependent Effects at the Extremes:** Does the impact of isolation remain constant across the lifespan, or does it diminish at extreme old age (e.g., > 85 years)? If the survival curves converge at late ages, what biological or sociological factors might explain this phenomenon?
5. **Gompertz Parameter Analysis:** Using the corrected data, fit the Gompertz law ($h(t) = h_0 e^{\alpha t}$). Does isolation primarily affect the rate of aging (α) or the damage load (η)?

Answer

1. Naive Analysis: An Incomplete Picture

Our initial naive analysis (standard Kaplan-Meier on pooled data) indicated that social isolation is indeed a risk factor. However, this pooled view assumes a uniform entry into the aging process. Given demographic differences in lifespan and living arrangements, we hypothesized this aggregate curve might be "dampened" by a survivor bias, potentially underestimating the true biological cost of isolation.

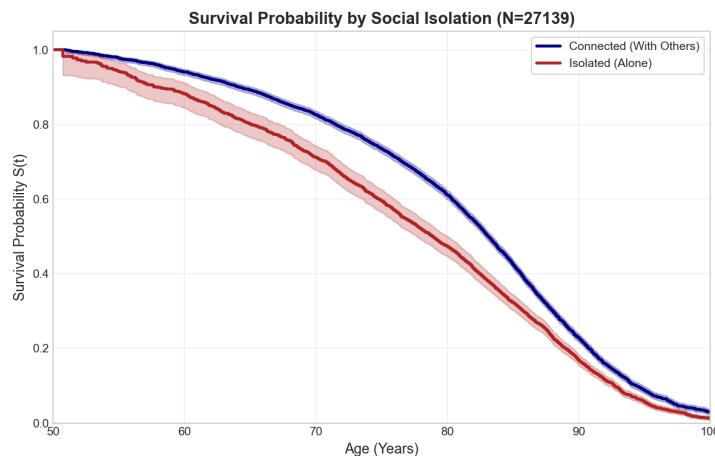


Figure 1: **Naive Kaplan-Meier survival curve (Pooled Population).** The isolated group (Red) shows reduced survival compared to the connected group (Blue), but this plot does not account for entry-age bias or sex differences.

2. Identifying Left Truncation Bias

To investigate potential biases, we analyzed the age distribution at study entry (Figure 2). Isolated females entered the study on average **6.4 years later** than connected females. A naive model grants these women "credit" for surviving decades before observation began. This artificial "immortal time" inflates the survival curve for the isolated group, masking the true risk. We applied **Left Truncation Correction** to adjust the risk set based on age of entry.

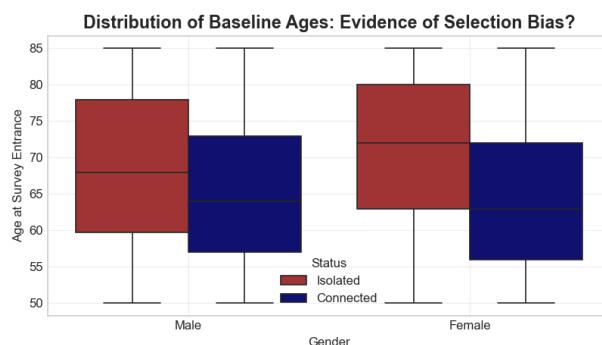


Figure 2: **Evidence of Survival Bias.** Isolated females are significantly older at baseline than connected females, creating a bias that must be corrected.

3. The Corrected Reality and Sex-Specific Vulnerability

Upon applying the correction, the true severity of isolation was revealed (Figure 3).

Observation: Social isolation reduces survival for **both** sexes, but the effect is drastically different in magnitude.

- **Males (High Vulnerability):** Isolation is catastrophic for men. The survival gap opens early and widens rapidly.
- **Females (Moderate Vulnerability):** The corrected curve confirms isolation is a risk, debunking the naive "protective" effect, but the gap is much smaller than in men.

Interpretation: This suggests a profound biological or sociological difference in response to isolation. Biologically, males may have a less robust physiological response to chronic psychosocial stress (e.g., higher inflammatory reactivity). Sociologically, "connection" often provides different benefits by sex; for older men, a spouse is often the primary caregiver and health manager. Losing that connection is a severe shock to health maintenance.

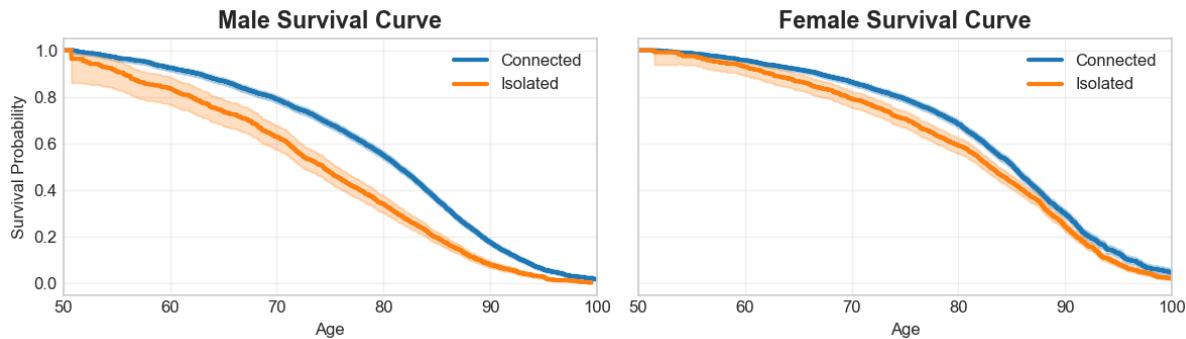


Figure 3: **Corrected Kaplan-Meier Analysis.** After accounting for entry age, isolation is a risk for both, but notably more severe in males (Left) than in females (Right).

4. Age-Dependent Effects at the Extremes

To visualize how the impact changes with age, we plotted the "Survival Advantage of Connection" ($S(t)_{Connected} - S(t)_{Isolated}$) over time (Figure 4).

Observation:

- **Males:** The advantage of connection is massive and sustained throughout old age.
- **Females:** The advantage is moderate in middle-old age but diminishes significantly after age 85, with the curves nearly converging toward 95.

Interpretation of Convergence at > 85 : At extreme old age, the nature of "connection" and "isolation" changes. 1. **Selection for Robustness:** Living alone at age 90 requires a high baseline of functional health (a high critical threshold X_c). These are functionally independent "Super Agers." 2. **Dependence of Connection:** Conversely, among the extremely elderly, living with others ("Connected") often signifies dependence due to frailty or cognitive decline, where family provides necessary care. Thus, at extreme ages, the biological damage of isolation may be counterbalanced by the fact that only the healthiest individuals remain capable of living alone.

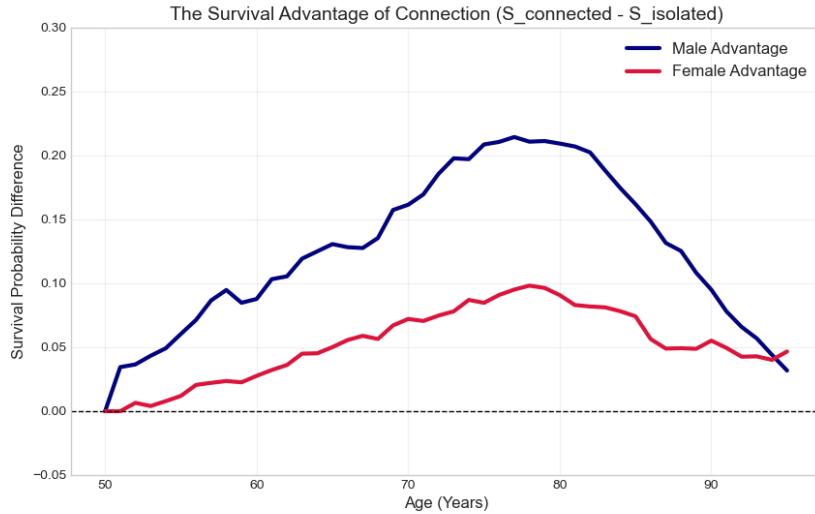


Figure 4: **The Survival Advantage of Connection over Time.** The plot shows the difference in survival probability ($S_{conn} - S_{iso}$). The gap is massive for males. For females, the gap narrows significantly after age 85, suggesting a convergence in mortality rates at extreme old age.

5. Gompertz Law and SR Model Interpretation

We performed a corrected Log-Cumulative Hazard analysis (Figure 5).

- **Scaling Effect:** The primary signature of isolation is an upward shift of the line (higher intercept h_0 , known as "Scaling") rather than a major change in the slope (α).
- **SR Model Interpretation:** A parallel shift implies an increase in the **Damage Production Rate** (η). Isolation acts as a constant metabolic stressor, adding a fixed load of damage per unit time, rather than accelerating the rate of system failure itself. The larger shift in men confirms the higher damage load they experience from isolation.

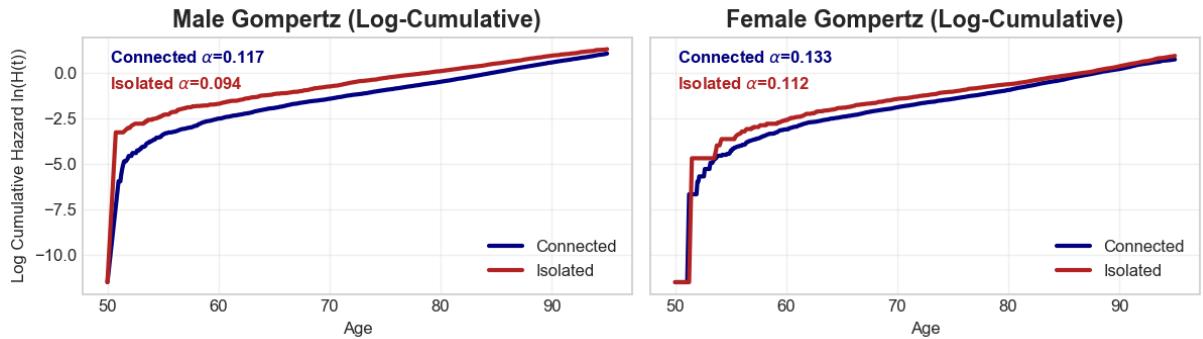


Figure 5: **Corrected Gompertz Analysis.** Both genders show "Scaling" (parallel shift), indicating increased damage production (η). The magnitude of the shift is larger in men.

Appendix: Python Code

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 from lifelines import KaplanMeierFitter, NelsonAalenFitter
6 from scipy.stats import linregress
7 import os
8
9 # =====
10 # 1. CONFIGURATION & SETUP
11 # =====
12 # Set visual style
13 plt.style.use('seaborn-v0_8-whitegrid')
14 sns.set_palette("deep")
15
16 # Directory and File Configuration
17 DATA_DIR = "DATA"
18 OUTPUT_DIR = "Figures"
19
20 # Create output directory if it doesn't exist
21 os.makedirs(OUTPUT_DIR, exist_ok=True)
22
23 FILE_PAIRS = [
24     ('DEMO_1999_2000.xpt', 'NHANES_1999_2000_MORT_2019_PUBLIC.dat'),
25     ('DEMO_2001_2002.xpt', 'NHANES_2001_2002_MORT_2019_PUBLIC.dat'),
26     ('DEMO_2003_2004.xpt', 'NHANES_2003_2004_MORT_2019_PUBLIC.dat'),
27     ('DEMO_2005_2006.xpt', 'NHANES_2005_2006_MORT_2019_PUBLIC.dat'),
28     ('DEMO_2007_2008.xpt', 'NHANES_2007_2008_MORT_2019_PUBLIC.dat'),
29     ('DEMO_2009_2010.xpt', 'NHANES_2009_2010_MORT_2019_PUBLIC.dat'),
30     ('DEMO_2011_2012.xpt', 'NHANES_2011_2012_MORT_2019_PUBLIC.dat'),
31     ('DEMO_2013_2014.xpt', 'NHANES_2013_2014_MORT_2019_PUBLIC.dat'),
32     ('DEMO_2015_2016.xpt', 'NHANES_2015_2016_MORT_2019_PUBLIC.dat'),
33     ('DEMO_2017_2018.xpt', 'NHANES_2017_2018_MORT_2019_PUBLIC.dat'),
34 ]
35
36 # Standardized Colors
37 COLORS = {
38     'Isolated': 'firebrick',
39     'Connected': 'navy'
40 }
41
42 # =====
43 # 2. DATA LOADING FUNCTIONS
44 # =====
45 def parse_mortality_file(filename):
46     """
47         Parses the fixed-width NHANES mortality file.
48         Extracts SEQN, MORTSTAT, and PERMTH_EXM.
49     """
50     filepath = os.path.join(DATA_DIR, filename)
51     data = []
52
53     if not os.path.exists(filepath):
54         print(f"Warning: Missing mortality file: {filepath}")
55         return pd.DataFrame()
56
57     with open(filepath, 'r') as f:
58         for line in f:
59             try:
60                 # SEQN: Bytes 0-6
61                 seqn = int(line[0:6])
```

```

62     # MORTSTAT: Byte 15
63     mortstat = int(line[15:16])
64     # PERMTH_EXM: Bytes 42-45
65     permth_str = line[42:45].strip()
66     permth = float(permth_str) if permth_str != '' else np.nan
67     data.append([seqn, mortstat, permth])
68 except ValueError:
69     continue
70
71 return pd.DataFrame(data, columns=['SEQN', 'mortstat', 'permth_exm'])
72
73 def load_and_merge_data(file_pairs):
74     """
75     Iterates through file pairs, loads SAS and DAT files, and merges them.
76     """
77     all_data = []
78     print(f"Starting data load from {DATA_DIR}...")
79
80     for demo_file, mort_file in file_pairs:
81         demo_path = os.path.join(DATA_DIR, demo_file)
82
83         if os.path.exists(demo_path):
84             try:
85                 # Load Demographics
86                 demo = pd.read_sas(demo_path)
87                 demo.columns = [c.upper() for c in demo.columns]
88
89                 # Select required columns
90                 cols_to_keep = ['SEQN', 'RIDAGEYR', 'RIAGENDR']
91                 if 'DMDHHSIZ' in demo.columns:
92                     cols_to_keep.append('DMDHHSIZ')
93
94                 # Check for missing columns (e.g., if variable names changed in
95                 # specific cycle)
96                 if not all(col in demo.columns for col in cols_to_keep):
97                     print(f"  Skipping {demo_file}: Missing required columns.")
98                     continue
99
100                demo = demo[cols_to_keep]
101
102                # Load Mortality
103                mort = parse_mortality_file(mort_file)
104
105                if not mort.empty:
106                    # Ensure join key is numeric
107                    demo['SEQN'] = pd.to_numeric(demo['SEQN'], errors='coerce')
108                    mort['SEQN'] = pd.to_numeric(mort['SEQN'], errors='coerce')
109
110                    merged = pd.merge(demo, mort, on='SEQN')
111                    all_data.append(merged)
112
113                except Exception as e:
114                    print(f"  Error reading {demo_file}: {e}")
115                else:
116                    print(f"  Missing demo file: {demo_path}")
117
118        if not all_data:
119            return pd.DataFrame()
120
121    return pd.concat(all_data, ignore_index=True)
122
123 # =====
124 # 3. DATA PROCESSING

```

```

124 # =====
125 raw_df = load_and_merge_data(FILE_PAIRS)
126
127 if raw_df.empty:
128     raise RuntimeError("No data loaded. Ensure 'DATA' folder contains the
129     correct files.")
130
131 # Remove duplicates
132 df = raw_df.drop_duplicates(subset=['SEQN'], keep='first')
133
134 # Type Conversion
135 numeric_cols = ['mortstat', 'permth_exm', 'RIDAGEYR', 'RIAGENDR', 'DMDHHSIZ']
136 for col in numeric_cols:
137     df[col] = pd.to_numeric(df[col], errors='coerce')
138
139 # Filter: Age 50+, remove invalid entries
140 df = df[df['RIDAGEYR'] >= 50].copy()
141 df = df.dropna(subset=numeric_cols)
142
143 # Define Social Groups (Household Size == 1 vs > 1)
144 df['Group'] = np.where(df['DMDHHSIZ'] == 1, 'Isolated', 'Connected')
145
146 # Decode Gender (1=Male, 2=Female in NHANES)
147 df['Gender'] = df['RIAGENDR'].map({1: 'Male', 2: 'Female'})
148
149 # Calculate Age at Event (for Left Truncation)
150 df['follow_up_years'] = df['permth_exm'] / 12.0
151 df['age_at_event'] = df['RIDAGEYR'] + df['follow_up_years']
152
153 print("=" * 30)
154 print(f"Data Processing Complete.")
155 print(f"Final N: {len(df)}")
156 print(df.groupby(['Gender', 'Group']).size())
157 print("=" * 30)
158
159 # =====
160 # 4. SURVIVAL ANALYSIS
161 # =====
162
163 # --- Figure 1: Overall Survival Probability ---
164 plt.figure(figsize=(10, 6), dpi=140)
165 kmf = KaplanMeierFitter()
166
167 plt.xlim(50, 100)
168 plt.ylim(0, 1.05)
169
170 for group in ['Connected', 'Isolated']:
171     mask = df['Group'] == group
172
173     kmf.fit(
174         durations=df.loc[mask, 'age_at_event'],
175         event_observed=df.loc[mask, 'mortstat'],
176         entry=df.loc[mask, 'RIDAGEYR'],
177         label=group
178     )
179     kmf.plot_survival_function(color=COLORS[group], linewidth=2.5)
180
181 plt.title(f'Overall Survival Probability by Social Isolation (N={len(df)})',
182           fontsize=14, fontweight='bold')
183 plt.xlabel('Age (Years)', fontsize=12)
184 plt.ylabel('Survival Probability S(t)', fontsize=12)
185 plt.grid(True, alpha=0.3)
186 plt.legend(frameon=True)

```

```

185 plt.savefig(os.path.join(OUTPUT_DIR, 'fig1.png'))
186
187
188 # --- Figure 2: Survival by Gender (Left Truncated) ---
189 fig, axes = plt.subplots(1, 2, sharey=True, figsize=(12, 6), dpi=120)
190
191 for i, gender in enumerate(['Male', 'Female']):
192     ax = axes[i]
193     df_g = df[df['Gender'] == gender]
194
195     for group in ['Connected', 'Isolated']:
196         mask = df_g['Group'] == group
197
198         kmf.fit(
199             durations=df_g.loc[mask, 'age_at_event'],
200             event_observed=df_g.loc[mask, 'mortstat'],
201             entry=df_g.loc[mask, 'RIDAGEYR'], # Handling left truncation
202             label=group
203         )
204         kmf.plot_survival_function(ax=ax, color=COLORS[group], linewidth=2.5)
205
206         ax.set_title(f'{gender} Survival Curve', fontsize=14, fontweight='bold')
207         ax.set_xlabel('Age')
208         if i == 0: ax.set_ylabel('Survival Probability')
209         ax.set_xlim(50, 100)
210         ax.grid(True, alpha=0.3)
211
212 plt.tight_layout()
213 plt.savefig(os.path.join(OUTPUT_DIR, 'fig2.png'))
214
215
216 # --- Figure 3: Gompertz Analysis (Log-Cumulative Hazard) ---
217 plt.figure(figsize=(14, 6))
218 naf = NelsonAalenFitter()
219 fig, axes = plt.subplots(1, 2, figsize=(12, 5), dpi=120, sharey=True)
220
221 start_age, end_age = 50, 95
222
223 for i, gender in enumerate(['Male', 'Female']):
224     ax = axes[i]
225     df_g = df[df['Gender'] == gender]
226
227     for group in ['Connected', 'Isolated']:
228         mask = df_g['Group'] == group
229
230         naf.fit(
231             durations=df_g.loc[mask, 'age_at_event'],
232             event_observed=df_g.loc[mask, 'mortstat'],
233             entry=df_g.loc[mask, 'RIDAGEYR'],
234             label=group
235         )
236
237         # Calculate Log Cumulative Hazard
238         H_t = naf.cumulative_hazard_
239         H_t = H_t.loc[start_age:end_age]
240         log_H_t = np.log(H_t + 1e-5)
241
242         ax.plot(log_H_t.index, log_H_t.values, color=COLORS[group], label=group,
243                 linewidth=2.5)
244
245         # Calculate Slope (Alpha) for annotation
246         if len(log_H_t) > 10:
247             x = log_H_t.index.values

```

```

247     y = log_H_t.iloc[:, 0].values
248     slope, _, _, _, _ = linregress(x, y)
249
250     # Annotate plot
251     y_pos = 0.9 - (0.1 if group == 'Isolated' else 0)
252     ax.text(0.05, y_pos, f'{group} $\backslash alpha$={slope:.3f}',
253             transform=ax.transAxes, color=COLORS[group], fontweight='bold')
254
255     ax.set_title(f'{gender} Gompertz Analysis', fontsize=14, fontweight='bold')
256     ax.set_xlabel('Age')
257     if i == 0: ax.set_ylabel('Log Cumulative Hazard ln(H(t))')
258     ax.legend()
259     ax.grid(True, alpha=0.3)
260
261 plt.tight_layout()
262 plt.savefig(os.path.join(OUTPUT_DIR, 'fig3.png'))
263
264
265
266 # --- Figure 4: Baseline Age Distribution ---
267 plt.figure(figsize=(8, 5), dpi=120)
268 sns.boxplot(
269     data=df, x='Gender', y='RIDAGEYR', hue='Group',
270     palette=COLORS
271 )
272
273 plt.title('Distribution of Baseline Ages', fontsize=14, fontweight='bold')
274 plt.ylabel('Age at Survey Entrance')
275 plt.xlabel('')
276 plt.grid(True, alpha=0.3)
277 plt.legend(title='Status')
278 plt.savefig(os.path.join(OUTPUT_DIR, 'fig4.png'))
279
280 # Print mean ages for verification
281 print("Mean Age by Group:")
282 print(df.groupby(['Gender', 'Group'])['RIDAGEYR'].mean())
283
284
285 # --- Figure 5: Survival Advantage (Difference Plot) ---
286 plt.figure(figsize=(10, 6))
287
288 timeline = np.arange(50, 96, 1)
289 gender_colors = {'Male': 'navy', 'Female': 'crimson'}
290
291 for gender in ['Male', 'Female']:
292     df_g = df[df['Gender'] == gender]
293
294     # 1. Fit Connected
295     mask_c = df_g['Group'] == 'Connected'
296     kmf_c = KaplanMeierFitter()
297     kmf_c.fit(
298         durations=df_g.loc[mask_c, 'age_at_event'],
299         event_observed=df_g.loc[mask_c, 'mortstat'],
300         entry=df_g.loc[mask_c, 'RIDAGEYR']
301     )
302     # Extract survival at specific timeline
303     # Note: survival_function_at_times returns a Series with the timeline as
304     # index
305     survival_c = kmf_c.survival_function_at_times(timeline)
306
307     # 2. Fit Isolated
308     mask_i = df_g['Group'] == 'Isolated'

```

```

308 kmf_i = KaplanMeierFitter()
309 kmf_i.fit(
310     durations=df_g.loc[mask_i, 'age_at_event'],
311     event_observed=df_g.loc[mask_i, 'mortstat'],
312     entry=df_g.loc[mask_i, 'RIDAGEYR']
313 )
314 survival_i = kmf_i.survival_function_at_times(timeline)
315
316 # 3. Calculate Difference
317 survival_difference = survival_c - survival_i
318
319 # 4. Plot
320 plt.plot(timeline, survival_difference, label=f'{gender} Advantage',
321           color=gender_colors[gender], linewidth=3)
322
323 plt.title('Survival Advantage of Connection (S_connected - S_isolated)',
324            fontsize=14, fontweight='bold')
324 plt.xlabel('Age (Years)', fontsize=12)
325 plt.ylabel('Survival Probability Difference', fontsize=12)
326 plt.axhline(0, color='black', linestyle='--', linewidth=1)
327 plt.legend(fontsize=12)
328 plt.grid(True, alpha=0.3)
329 plt.ylim(-0.05, 0.30)
330 plt.savefig(os.path.join(OUTPUT_DIR, 'fig5.png'))
331
332 print("Analysis Complete.")

```