Our Modelling Challenge

Diart, Gyula, Stan, Calvin

Colchester Royal Grammar School

Team #18907

What technical computing meant to us

- ❏ Theory → real-life impact

What technical computing meant to us

- ❑ Theory → real-life impact
  - ❑ Abstraction!

What technical computing meant to us

- Theory → real-life impact
  - Abstraction!
- This problem was not one equation...

What technical computing meant to us

- Theory → real-life impact
  - Abstraction!
- This problem was not one equation…
  - We had 3 scales

What technical computing meant to us

- ❑ Theory → real-life impact
  - ❑ Abstraction!
- ❑ This problem was not one equation…
  - ❑ We had 3 scales
    - ❑ Household affordability

What technical computing meant to us

- Theory → real-life impact
  - Abstraction!
- This problem was not one equation...
  - We had 3 scales
    - Household affordability
    - Behavioural gambling dynamics

What technical computing meant to us

- ❏ Theory → real-life impact
  - ❏ Abstraction!
- ❏ This problem was not one equation…
  - ❏ We had 3 scales
    - ❏ Household affordability
    - ❏ Behavioural gambling dynamics
    - ❏ Long-run recovery models

What technical computing meant to us

- Theory → real-life impact
  - Abstraction!
- This problem was not one equation…
  - We had 3 scales
    - Household affordability
    - Behavioural gambling dynamics
    - Long-run recovery models

Affordability → Behavioural escalation → Recovery burden

How the models connect

How the models connect

- ❑ Question 1

How the models connect

- ❑ Question 1
  - ❑ identifies vulnerability before gambling begins

How the models connect

- ❑ Question 1
  - ❑ identifies vulnerability before gambling begins
- ❑ Question 2

How the models connect

- ❑ Question 1
  - ❑ identifies vulnerability before gambling begins
- ❑ Question 2
  - ❑ models how behaviour converts vulnerability into losses

How the models connect

- Question 1
  - identifies vulnerability before gambling begins
- Question 2
  - models how behaviour converts vulnerability into losses
- Question 3

How the models connect

- ❑ Question 1
  - ❑ identifies vulnerability before gambling begins
- ❑ Question 2
  - ❑ models how behaviour converts vulnerability into losses
- ❑ Question 3
  - ❑ measures how long the damage lasts after the betting stops

Playing with House Money

Q1 - an introduction

Q1 - an introduction

- ❏ Model a person’s disposable income

Q1 - an introduction

- ❑ Model a person’s disposable income
    - ❑ From salary and demographics

Q1 - an introduction

- ❏ Model a person’s disposable income
  - ❏ From salary and demographics
  - ❏ Test how vulnerable it is to unexpected emergency costs

Q1 - an introduction

- Model a person’s disposable income
    - From salary and demographics
    - Test how vulnerable it is to unexpected emergency costs
- We built a 2-part model:

Q1 - an introduction

- Model a person’s disposable income
  - From salary and demographics
  - Test how vulnerable it is to unexpected emergency costs
- We built a 2-part model:
  - Deterministic layer

Q1 - an introduction

- Model a person’s disposable income
  - From salary and demographics
  - Test how vulnerable it is to unexpected emergency costs
- We built a 2-part model:
  - Deterministic layer
  - Stochastic layer

Simplified Overview

$$Y_d = \sum_{m=0}^{11} \max\left(Y_t - EQ_{ind}\left(C + V(m)\right) - S_m, 0\right)$$

Simplified Overview

$$Y_d = \sum_{m=0}^{11} \max\left(\boxed{Y_t} - EQ_{ind}\left(C + V\left(m\right)\right) - S_m, 0\right)$$

Net Income

$$Y_t = \frac{Y\left(1-\tau\right)}{12}$$

Simplified Overview

$$Y_d = \sum_{m=0}^{11} \max\left(\boxed{Y_t} - \boxed{EQ_{ind}\left(C + V(m)\right)} - S_m, 0\right)$$

Net Income

$$Y_t = \frac{Y(1-\tau)}{12}$$

Essential Costs

$$EQ_{ind} = \frac{1 + 0.5(a-1) + 0.3c}{a}$$

$$V(m) = U_0 + A\cos\left(\frac{2\pi}{12}m\right)$$

$$C = H + F + T + M$$

$$M = \begin{cases} M_0 & \text{if } \alpha \leq 65 \\ 2.5M_0 & \text{if } \alpha > 65 \end{cases}$$

Simplified Overview

$$Y_d = \sum_{m=0}^{11} \max\left(\boxed{Y_t} - \boxed{EQ_{ind}\left(C + V(m)\right)} - \boxed{S_m}, 0\right)$$

Net Income

$$Y_t = \frac{Y(1-\tau)}{12}$$

Essential Costs

$$EQ_{ind} = \frac{1 + 0.5(a-1) + 0.3c}{a}$$

$$V(m) = U_0 + A\cos\left(\frac{2\pi}{12}m\right)$$

$$C = H + F + T + M$$

$$M = \begin{cases} M_0 & \text{if } \alpha \leq 65 \\ 2.5M_0 & \text{if } \alpha > 65 \end{cases}$$

Emergency Costs

$$S_m = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{n_i} s_{i,j,m}$$

$$s_{i,j} \sim \mathrm{LogNormal}(\mu, \sigma^2), \quad j = 1, 2, \ldots, n_i$$

$$n_i \sim \mathrm{Poisson}(\lambda), \quad i = 1, 2, \ldots, N$$

$$\lambda = \lambda_0 \cdot \lambda_{age} \cdot \lambda_{size}$$

$$\lambda_{size} = 1.0 + 0.2\,(a + c - 1)$$

$$\lambda_{age} = \begin{cases} 1 & \text{if } \alpha \leq 40 \\ 1.2 & \text{if } \alpha > 40 \end{cases}$$

What data did we use?

What data did we use?

- ❑ Tax

What data did we use?

- Tax
    - IRS, UK Gov and States

What data did we use?

- ❑ Tax
  - ❑ IRS, UK Gov and States
- ❑ Cost of Living

What data did we use?

- ❑ Tax
  - ❑ IRS, UK Gov and States
- ❑ Cost of Living
  - ❑ ONS, MIT Living Wage Calculator, Research Journals

Stan

What data did we use?

- ❑ Tax
  - ❑ IRS, UK Gov and States
- ❑ Cost of Living
  - ❑ ONS, MIT Living Wage Calculator, Research Journals
- ❑ How we used the data

What data did we use?

- Tax
  - IRS, UK Gov and States
- Cost of Living
  - ONS, MIT Living Wage Calculator, Research Journals
- How we used the data
  - Parsed deterministic Cost of Living into a CSV file (with Python scripts)

Stan

What data did we use?

- Tax
  - IRS, UK Gov and States
- Cost of Living
  - ONS, MIT Living Wage Calculator, Research Journals
- How we used the data
  - Parsed deterministic Cost of Living into a CSV file (with Python scripts)
  - Parameters of shock distribution and intensity calculated from research data

Data used

Data used

1 - Tax Rates

	A	B	C
	Country	**Region**	**Effective_State_Tax_Rate**
	Categorical	Text	Number
1	Country	Region	Effective_State_Tax_Rate
2	US	Alabama	0.05
3	US	Alaska	0
4	US	Arizona	0.025
5	US	Arkansas	0.039
6	US	California	0.133
7	US	Colorado	0.044
8	US	Connecticut	0.0699
9	US	Delaware	0.066
10	US	Florida	0
11	US	Georgia	0.0519
12	US	Hawaii	0.11
13	US	Idaho	0.053
14	US	Illinois	0.0495
15	US	Indiana	0.0295
16	US	Iowa	0.038
17	US	Kansas	0.0558
18	US	Kentucky	0.035
19	US	Louisiana	0.03
20	US	Maine	0.0715
21	US	Maryland	0.065
22	US	Massachus...	0.09
23	US	Michigan	0.0425

Data used

1 - Tax Rates

	A	B	C
	Country	Region	Effective_State_Tax_Rate
	Categorical	Text	Number
1	Country	Region	Effective_State_Tax_Rate
2	US	Alabama	0.05
3	US	Alaska	0
4	US	Arizona	0.025
5	US	Arkansas	0.039
6	US	California	0.133
7	US	Colorado	0.044
8	US	Connecticut	0.0699
9	US	Delaware	0.066
10	US	Florida	0
11	US	Georgia	0.0519
12	US	Hawaii	0.11
13	US	Idaho	0.053
14	US	Illinois	0.0495
15	US	Indiana	0.0295
16	US	Iowa	0.038
17	US	Kansas	0.0558
18	US	Kentucky	0.035
19	US	Louisiana	0.03
20	US	Maine	0.0715
21	US	Maryland	0.065
22	US	Massachus...	0.09
23	US	Michigan	0.0425

2 - Essential Costs

	A	B	C	D	E	F
	Country	Region	Base_Housing_Annual	Base_Food_Annual	Base_Transport_Annual	Base_Healthcare_Annual
	Categorical	Text	Number	Number	Number	Number
1	Country	Region	Base_Housing_Annual	Base_Food_Annual	Base_Transport_Annual	Base_Healthcare_Annual
2	US	Texas	14744	3856	8454	3271
3	US	California	22471	4463	8681	3255
4	US	New York	35525	7116	4710	4695
5	US	Florida	22049	4802	7395	2801
6	UK	England	2769	1706	2101	250
7	UK	Wales	2166	1617	2114	250
8	UK	Scotland	2127	1599	1973	250
9	UK	Northern Ireland	1854	2025	2327	250

Model Structure

(1) Create the ‘personas’

```
personas = struct(...
    'Description', { ...
        'Young Single, TX (US)', ...
        'Young Single, CA (US)', ...
        'Mid-Career Single, NY (US)', ...
        'Mid-Career Family, NY (US)', ...
        'Pre-Retiree, FL (US)', ...
        'Young Single, England (UK)', ...
        'Mid-Career Family, Scotland (UK)', ...
        'Pensioner, Wales (UK)' ...
    }, ...
    'Salary',   {35000, 35000, 120000, 120000, 65000,  35000,   65000,    25000}, ...
    'Age',      {   24,    24,     40,     40,    62,     28,      45,       72}, ...
    'Country',  { 'US',  'US',   'US',   'US',  'US',   'UK',    'UK',     'UK'}, ...
    'Region',   {'Texas', 'California', 'New York', 'New York', 'Florida', 'England', 'Scotland', 'Wales'}, ...
    'Adults',   {    1,     1,      1,      2,     2,      1,       2,        1}, ...
    'Children', {    0,     0,      0,      2,     0,      0,       2,        0} ...
);
```

Model Structure

Stan

(1) Create the ‘Personas’

```matlab
% 2. Define the Demonstration Personas
personas = struct(...
    'Description', { ...
        'Young Single, TX (US)', ...
        'Young Single, CA (US)', ...
        'Mid-Career Single, NY (US)', ...
        'Mid-Career Family, NY (US)', ...
        'Pre-Retiree, FL (US)', ...
        'Young Single, England (UK)', ...
        'Mid-Career Family, Scotland (UK)', ...
        'Pensioner, Wales (UK)' ...
    }, ...
    'Salary',   {35000, 35000, 120000, 120000, 65000,  35000,   65000,    25000}, ...
    'Age',      {   24,    24,     40,     40,    62,     28,      45,       72}, ...
    'Country',  { 'US',  'US',   'US',   'US',  'US',   'UK',    'UK',     'UK'}, ...
    'Region',   {'Texas', 'California', 'New York', 'New York', 'Florida', 'England', 'Scotland', 'Wales'}, ...
    'Adults',   {    1,     1,      1,      2,     2,      1,       2,        1}, ...
    'Children', {    0,     0,      0,      2,     0,      0,       2,        0} ...
);
```

(2) Apply Taxes on ‘Personas’

```matlab
function Net_Income = calculate_taxes(persona)
% CALCULATE_TAXES Computes exact post-tax net income based on
% 2024 U.S. and U.K. statutory tax brackets.
%
% Inputs:
%   persona : Struct containing demographic and gross salary data
%             (Salary, Age, Country, Region, Adults, Children)
%
% Outputs:
%   Net_Income : Final post-tax Net Income

    % Initialize Gross Salary
    Salary = persona.Salary;
    Country = persona.Country;

    if strcmpi(Country, 'US')
        % --- US Tax Calculation ---

        % 1. Standard Deduction
        if persona.Adults > 1
            std_deduction = 29200; % Married standard deduction
        else
            std_deduction = 14600; % Single standard deduction
        end

        taxable_income = max(Salary - std_deduction, 0);

        % 2. 2024 Progressive Marginal Tax Brackets
        % Hardcoded the 10%, 12%, 22%, 24%, 32%, 35%, 37% brackets
        if persona.Adults > 1
            % Married Filing Jointly limits
            brackets = [23200, 94300, 201050, 383900, 487450, 768400, inf];
        else
            % Single limits
            brackets = [11600, 47150, 100525, 191950, 243725, 609350, inf];
        end
        rates = [0.10, 0.12, 0.22, 0.24, 0.32, 0.35, 0.37];

        federal_tax = 0;
        prev_limit = 0;
        for i = 1:length(rates)
            if taxable_income > prev_limit
                taxable_in_bracket = min(taxable_income, brackets(i)) - prev_limit;
                federal_tax = federal_tax + (taxable_in_bracket * rates(i));
                prev_limit = brackets(i);
            else
                break;
            end
        end

        % 3. FICA (Payroll Tax)
        ss_tax = min(Salary, 168600) * 0.062; % Social Security
        medicare_tax = Salary * 0.0145;       % Medicare
        fica_tax = ss_tax + medicare_tax;

        % 4. State Tax
        % Look up in us_state_taxes.csv / tax_rates.csv
        state_tax = 0;

        persistent state_data;
        if isempty(state_data)
            if exist('Q1/data/tax_rates.csv', 'file')
                state_data = readtable('Q1/data/tax_rates.csv', 'VariableNamingRule', 'preserve');
            else
                error('calculate_taxes:MissingData', 'Could not finding state tax data file.');
            end
        end

        % Try to find region match
        is_rr = strcmpi(string(state_data.Region), string(persona.Region));
        if any(is_rr)
            match_row = state_data(is_rr, :);
            effective_rate = match_row.Effective_State_Tax_Rate(1);
            state_tax = Salary * effective_rate;
        end

        % Net Income US
        Net_Income = Salary - (federal_tax + fica_tax + state_tax);

    elseif strcmpi(Country, 'UK')
        % --- UK Tax Calculation ---

        % 1. Personal Allowance
        % Reduce by £1 for every £2 earned over £100,000
        reduction = max(0, (Salary - 100000) / 2);
        pa = max(0, 12570 - reduction);

        taxable_income = max(Salary - pa, 0);

        % 2. Income Tax Brackets (2024/25)
        if strcmpi(persona.Region, 'Scotland')
            % Scotland 2024-25 Income Tax (taxable income ranges)
            % Starter: 19% on £1 - £2,297
            % Basic: 20% on £2,298 - £13,991
            % Intermediate: 21% on £13,992 - £31,092
            % Higher: 42% on £31,093 - £62,430
            % Advanced: 45% on £62,431 - £125,140
            % Top: 48% over £125,140
            limits = [2297, 13991, 31092, 62430, 125140, inf];
            rates  = [0.19, 0.20, 0.21, 0.42, 0.45, 0.48];
        else
            % Rest of UK (England, Wales, NI) 2024-25 Income Tax
            % Basic: 20% on £1 - £37,700
            % Higher: 40% on £37,701 - £125,140
            % Additional: 45% over £125,140
            limits = [37700, 125140, inf];
            rates  = [0.20, 0.40, 0.45];
        end

        income_tax = 0;
        prev_limit = 0;
        for i = 1:length(rates)
            if taxable_income > prev_limit
                taxable_in_bracket = min(taxable_income, limits(i)) - prev_limit;
                income_tax = income_tax + (taxable_in_bracket * rates(i));
                prev_limit = limits(i);
            else
                break;
            end
        end

        % 3. National Insurance (Class 1)
        % 0% up to £12,570; 8% between £12,570 and £50,270; 2% above £50,270
        ni_8 = 0.08 * max(0, min(Salary, 50270) - 12570);
        ni_2 = 0.02 * max(0, Salary - 50270);

        NI = ni_8 + ni_2;

        % Net Income UK
        Net_Income = Salary - (income_tax + NI);

    else
        error('calculate_taxes:UnknownCountry', ...
            'Country "%s" not supported.', Country);
    end

end
```

$\longleftrightarrow$

Tax Rates

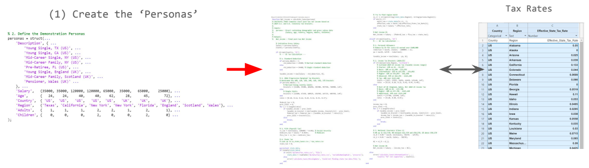
	A	B	C
	Country	Region	Effective_State_Tax_Rate
	Categorical	Text	Number
1	Country	Region	Effective_State_Tax_Rate
2	US	Alabama	0.05
3	US	Alaska	0
4	US	Arizona	0.025
5	US	Arkansas	0.039
6	US	California	0.133
7	US	Colorado	0.044
8	US	Connecticut	0.0699
9	US	Delaware	0.066
10	US	Florida	0
11	US	Georgia	0.0519
12	US	Hawaii	0.11
13	US	Idaho	0.053
14	US	Illinois	0.0495
15	US	Indiana	0.0295
16	US	Iowa	0.038
17	US	Kansas	0.0558
18	US	Kentucky	0.035
19	US	Louisiana	0.03
20	US	Maine	0.0715
21	US	Maryland	0.065
22	US	Massachus…	0.09
23	US	Michigan	0.0425

$$Y_t = \frac{Y(1-\tau)}{12}$$

Symbol	Variable
Y_t	Monthly Net Income
Y	Gross Income
τ	Tax Rate

Model Structure

Stan



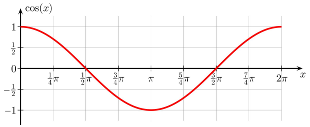
(3) Calculate Essential Costs - Part 1

```
function base_essentials = calculate_essentials(persona, cost_data)
% CALCULATE_ESSENTIALS Estimates baseline single-adult essential costs via table
% lookups from empirical cost data.
%
% Inputs:
%   persona   : Struct with fields (Salary, Age, Country, Region, Adults, Children)
%   cost_data : MATLAB table loaded from essential_costs.csv
%
% Output:
%   base_essentials : Baseline essential costs for a single adult.

    % Filter the cost_data table to match the persona.Country and persona.Region
    is_cc = strcmpi(string(cost_data.Country), string(persona.Country));
    is_rr = strcmpi(string(cost_data.Region), string(persona.Region));

    match_row = cost_data(is_cc & is_rr, :);

    if isempty(match_row)
        error('calculate_essentials:NotFound', ...
            'Could not find matching cost data for Country: %s, Region: %s', ...
            persona.Country, persona.Region);
    end

    % If there are multiple matches, take the first one
    match_row = match_row(1, :);

    % Extract the annual base costs
    Base_Housing    = match_row.Base_Housing_Annual;
    Base_Food       = match_row.Base_Food_Annual;
    Base_Transport  = match_row.Base_Transport_Annual;
    Base_Healthcare = match_row.Base_Healthcare_Annual;

    % Healthcare Age Scaling:
    % If persona.Age > 65 and Country == 'US', multiply Base_Healthcare by 2.5
    if persona.Age > 65 && strcmpi(persona.Country, 'US')
        Base_Healthcare = Base_Healthcare * 2.5;
    end

    % Compute Utility Cost using a sinusoidal model peaking in winter (January, month 1)
    % Assume average annual utility is base rate, shifting with cosine
    months = 1:12;
    if strcmpi(persona.Country, 'US')
        annual_avg_utility = 2400; % Average utility cost per year ($200/month)
        annual_amp_utility = 1200; % Swing amplitude (+/- $100/month difference in winter/summer)
    else
        annual_avg_utility = 1800; % Average utility cost per year (£150/month)
        annual_amp_utility = 900;  % Swing amplitude (+/- £75/month difference)
    end

    % Annualized utility modeled sinusoidally. Peak in Month 1 (January, winter heating).
    annualized_utility = annual_avg_utility + annual_amp_utility * cos(2 * pi * (months - 1) / 12);

    % Compute Total Base Essentials for a Single Adult (Now a 1x12 array across months)
    base_essentials = Base_Housing + Base_Food + Base_Transport + Base_Healthcare + annualized_utility;
end
```

Essential Costs

	A	B	C	D	E	F
	Country	Region	Base_Housing_Annual	Base_Food_Annual	Base_Transport_Annual	Base_Healthcare_Annual
	Categorical	Text	Number	Number	Number	Number
1	Country	Region	Base_Housing_Annual	Base_Food_Annual	Base_Transport_Annual	Base_Healthcare_Annual
2	US	Texas	14744	3856	8454	3271
3	US	California	22471	4463	8681	3255
4	US	New York	35525	7116	4710	4695
5	US	Florida	22049	4802	7395	2801
6	UK	England	2769	1706	2101	250
7	UK	Wales	2166	1617	2114	250
8	UK	Scotland	2127	1599	1973	250
9	UK	Northern Ireland	1854	2025	2327	250

+



$$C = H + F + T + M \qquad M = \begin{cases} M_0 & \text{if } \alpha \leq 65 \\ 2.5M_0 & \text{if } \alpha > 65 \end{cases}$$

Symbol	Variable
C	Fixed Costs
H	Housing Costs
F	Food Costs
T	Transportation
M	Medical Costs
M_0	Base Medical Costs
α	Age

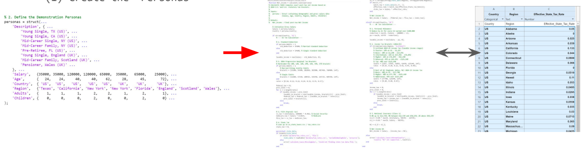
Model Structure

Stan

(1) Create the 'Personas'

(2) Apply Taxes on 'Personas'

Tax Rates



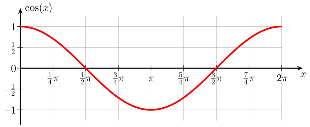
(3) Calculate Essential Costs - Part 2

```
function base_essentials = calculate_essentials(persona, cost_data)
% CALCULATE_ESSENTIALS Estimates baseline single-adult essential costs via table
% lookups from empirical cost data.
%
% Inputs:
%   persona   : Struct with fields (Salary, Age, Country, Region, Adults, Children)
%   cost_data : MATLAB table loaded from essential_costs.csv
%
% Output:
%   base_essentials : Baseline essential costs for a single adult.

    % Filter the cost_data table to match the persona.Country and persona.Region
    is_cc = strcmpi(string(cost_data.Country), string(persona.Country));
    is_rr = strcmpi(string(cost_data.Region), string(persona.Region));

    match_row = cost_data(is_cc & is_rr, :);

    if isempty(match_row)
        error('calculate_essentials:NotFound', ...
            'Could not find matching cost data for Country: %s, Region: %s', ...
            persona.Country, persona.Region);
    end

    % If there are multiple matches, take the first one
    match_row = match_row(1, :);

    % Extract the annual base costs
    Base_Housing    = match_row.Base_Housing_Annual;
    Base_Food       = match_row.Base_Food_Annual;
    Base_Transport  = match_row.Base_Transport_Annual;
    Base_Healthcare = match_row.Base_Healthcare_Annual;

    % Healthcare Age Scaling:
    % If persona.Age > 65 and Country == 'US', multiply Base_Healthcare by 2.5
    if persona.Age > 65 && strcmpi(persona.Country, 'US')
        Base_Healthcare = Base_Healthcare * 2.5;
    end

    % Compute Utility Cost using a sinusoidal model peaking in winter (January, month 1)
    % Assume average annual utility is base rate, shifting with cosine
    months = 1:12;
    if strcmpi(persona.Country, 'US')
        annual_avg_utility = 2400; % Average utility cost per year ($200/month)
        annual_amp_utility = 1200; % Swing amplitude (+/- $100/month difference in winter/summer)
    else
        annual_avg_utility = 1800; % Average utility cost per year (£150/month)
        annual_amp_utility = 900;  % Swing amplitude (+/- £75/month difference)
    end

    % Annualized utility modeled sinusoidally. Peak in Month 1 (January, winter heating).
    annualized_utility = annual_avg_utility + annual_amp_utility * cos(2 * pi * (months - 1) / 12);

    % Compute Total Base Essentials for a Single Adult (Now a 1x12 array across months)
    base_essentials = Base_Housing + Base_Food + Base_Transport + Base_Healthcare + annualized_utility;
end
```

Essential Costs

	A	B	C	D	E	F
	Country	Region	Base_Housing_Annual	Base_Food_Annual	Base_Transport_Annual	Base_Healthcare_Annual
	Categorical	Text	Number	Number	Number	Number
1	Country	Region	Base_Housing_Annual	Base_Food_Annual	Base_Transport_Annual	Base_Healthcare_Annual
2	US	Texas	14744	3856	8454	3271
3	US	California	22471	4463	8681	3255
4	US	New York	35525	7116	4710	4695
5	US	Florida	22049	4802	7395	2801
6	UK	England	2769	1706	2101	250
7	UK	Wales	2166	1617	2114	250
8	UK	Scotland	2127	1599	1973	250
9	UK	Northern Ireland	1854	2025	2327	250

+

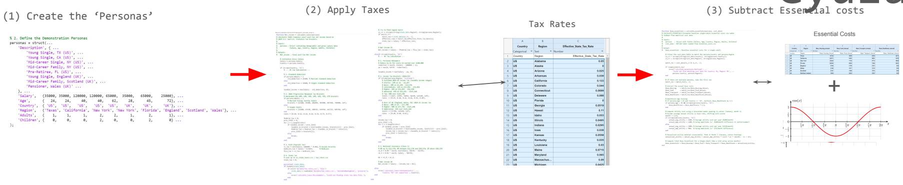


$$V(m) = U_0 + A\cos(\frac{2\pi}{12}m)$$

Symbol	Variable
$V(m)$	Monthly Variable Costs
U_0	Base Utility Costs
A	Variation Amplitude
m	Month (from 0 to 11)

Gyula

Model Structure



(4) Apply Household Structure

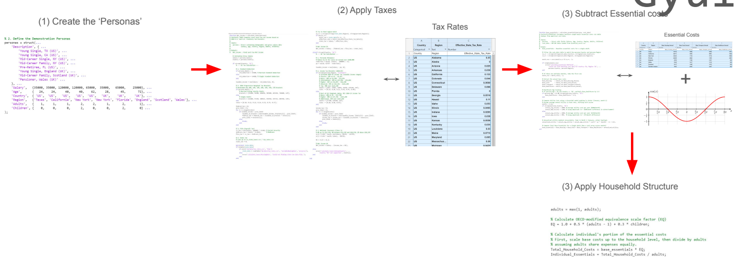
```
adults = max(1, adults);

% Calculate OECD-modified equivalence scale factor (EQ)
EQ = 1.0 + 0.5 * (adults - 1) + 0.3 * children;

% Calculate individual's portion of the essential costs
% First, scale base costs up to the household level, then divide by adults
% assuming adults share expenses equally.
Total_Household_Costs = base_essentials * EQ;
Individual_Essentials = Total_Household_Costs / adults;
```

$$EQ_{ind} = \frac{1 + 0.5(a - 1) + 0.3c}{a}$$

Symbol	Variable
EQ_{ind}	OECD-modified equivalence scale
a	Number of Adults
c	Number of Child

Gyula

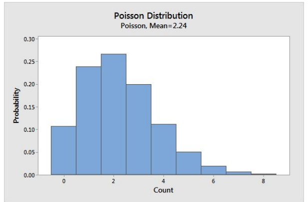
Model Structure



(5) Apply Financial Shocks - Part 1

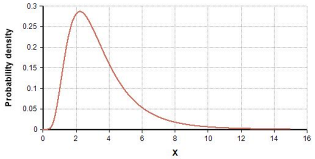
```
function [results] = simulate_monthly_shocks(monthly_disposable_income, age, country, adults, children, num_sims)
% SIMULATE_MONTHLY_SHOCKS Simulates the impact of random financial shocks
% using a Frequency-Severity actuarial framework.
%
% Inputs:
%   monthly_disposable_income - Baseline monthly disposable income
%   age                       - Age of the primary earner
%   country                   - Country of residence ('US' or 'UK')
%   adults                    - Number of adults in the household
%   children                  - Number of children in the household
%   num_sims                  - (Optional) Number of simulations to run. Default: 10000.
%
% Outputs:
%   results.Base_Monthly_Income        - Deterministic monthly disposable income
%   results.Median_Monthly_Income      - Median of simulated monthly incomes
%   results.Worst_5th_Percentile_Month - 5th percentile of simulated monthly incomes
%   results.Prob_of_Debt_Percent       - Percentage of simulations resulting in debt

    % Set default number of simulations if not provided
    if nargin < 6 || isempty(num_sims)
        num_sims = 10000;
    end

    % Step A: Determine Severity Parameters (Log-Normal Distribution)
    country = upper(country); % Ensure case-insensitivity
    if strcmp(country, 'US')
        mu = 7.6;
        sigma = 0.8;
    elseif strcmp(country, 'UK')
        mu = 6.48;
        sigma = 0.75;
    else
        error('Invalid country. Must be ''US'' or ''UK''.');
    end

    % Step B: Determine Frequency Parameter (Poisson Distribution)
    base_lambda_monthly = 0.0763;

    % Calculate age_mod
    if age > 40
        age_mod = 1.2;
    else
        age_mod = 1.0;
    end

    % Calculate size_mod
    size_mod = 1.0 + 0.2 * ((adults + children) - 1);

    % Final monthly lambda
    lambda_monthly = base_lambda_monthly * age_mod * size_mod;

    % Step C: The Monte Carlo Simulation (Vectorized for Speed)
    % Initialize an array simulated_monthly_incomes of size num_sims x 1
    simulated_monthly_incomes = zeros(num_sims, 1);

    % Generate the number of emergency events for all universes at once
    N = poissrnd(lambda_monthly, num_sims, 1);

    for i = 1:num_sims
        if N(i) == 0
            shock_cost = 0;
        elseif N(i) > 0
            % Draw N(i) random samples from Log-Normal and sum them
            shock_cost = sum(lognrnd(mu, sigma, N(i), 1));
        end
        simulated_monthly_incomes(i) = monthly_disposable_income - shock_cost;
    end

    % Step D: Return Risk Metrics
    % Store the results in a struct
    results.Base_Monthly_Income = monthly_disposable_income;
    results.Median_Monthly_Income = median(simulated_monthly_incomes);
    results.Worst_5th_Percentile_Month = prctile(simulated_monthly_incomes, 5);

    % Calculate probability of debt as a percentage
    prob_debt = sum(simulated_monthly_incomes < 0) / num_sims;
    results.Prob_of_Debt_Percent = prob_debt * 100;

end
```

↔



+

Log-normal Distribution



$$\lambda = \lambda_0 \cdot \lambda_{age} \cdot \lambda_{size}$$

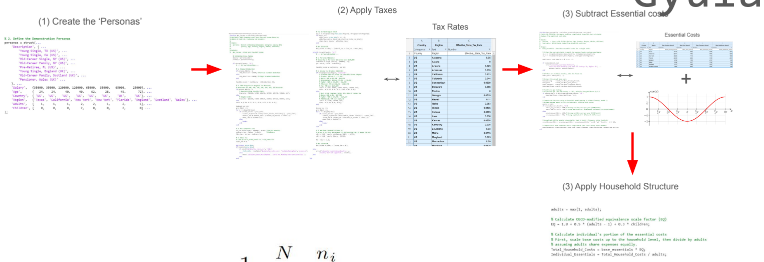
$$\lambda_{size} = 1.0 + 0.2\,(a + c - 1)$$

$$\lambda_{age} = \begin{cases} 1 & \text{if } \alpha \leq 40 \\ 1.2 & \text{if } \alpha > 40 \end{cases}$$

Symbol	Variable
λ	Mean Shock Rate
λ_0	Base Rate
λ_{size}	Size Rate Modifier
λ_{age}	Age Rate Modifier
α	Age

Gyula

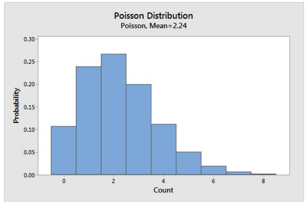
Model Structure



(5) Apply Financial Shocks - Part 2

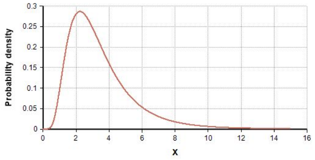
```
function [results] = simulate_monthly_shocks(monthly_disposable_income, age, country, adults, children, num_sims)
% SIMULATE_MONTHLY_SHOCKS Simulates the impact of random financial shocks
% using a Frequency-Severity actuarial framework.
%
% Inputs:
%   monthly_disposable_income - Baseline monthly disposable income
%   age                       - Age of the primary earner
%   country                   - Country of residence ('US' or 'UK')
%   adults                    - Number of adults in the household
%   children                  - Number of children in the household
%   num_sims                  - (Optional) Number of simulations to run. Default: 10000.
%
% Outputs:
%   results.Base_Monthly_Income        - Deterministic monthly disposable income
%   results.Median_Monthly_Income      - Median of simulated monthly incomes
%   results.Worst_5th_Percentile_Month - 5th percentile of simulated monthly incomes
%   results.Prob_of_Debt_Percent       - Percentage of simulations resulting in debt

    % Set default number of simulations if not provided
    if nargin < 6 || isempty(num_sims)
        num_sims = 10000;
    end

    % Step A: Determine Severity Parameters (Log-Normal Distribution)
    country = upper(country); % Ensure case-insensitivity
    if strcmp(country, 'US')
        mu = 7.6;
        sigma = 0.8;
    elseif strcmp(country, 'UK')
        mu = 6.48;
        sigma = 0.75;
    else
        error('Invalid country. Must be ''US'' or ''UK''.');
    end

    % Step B: Determine Frequency Parameter (Poisson Distribution)
    base_lambda_monthly = 0.0763;

    % Calculate age_mod
    if age > 40
        age_mod = 1.2;
    else
        age_mod = 1.0;
    end

    % Calculate size_mod
    size_mod = 1.0 + 0.2 * ((adults + children) - 1);

    % Final monthly lambda
    lambda_monthly = base_lambda_monthly * age_mod * size_mod;

    % Step C: The Monte Carlo Simulation (Vectorized for Speed)
    % Initialize an array simulated_monthly_incomes of size num_sims x 1
    simulated_monthly_incomes = zeros(num_sims, 1);

    % Generate the number of emergency events for all universes at once
    N = poissrnd(lambda_monthly, num_sims, 1);

    for i = 1:num_sims
        if N(i) == 0
            shock_cost = 0;
        elseif N(i) > 0
            % Draw N(i) random samples from Log-Normal and sum them
            shock_cost = sum(lognrnd(mu, sigma, N(i), 1));
        end
        simulated_monthly_incomes(i) = monthly_disposable_income - shock_cost;
    end

    % Step D: Return Risk Metrics
    % Store the results in a struct
    results.Base_Monthly_Income = monthly_disposable_income;
    results.Median_Monthly_Income = median(simulated_monthly_incomes);
    results.Worst_5th_Percentile_Month = prctile(simulated_monthly_incomes, 5);

    % Calculate probability of debt as a percentage
    prob_debt = sum(simulated_monthly_incomes < 0) / num_sims;
    results.Prob_of_Debt_Percent = prob_debt * 100;

end
```

$\longleftrightarrow$



+

Log-normal Distribution



$$S_m = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{n_i} s_{i,j,m}$$

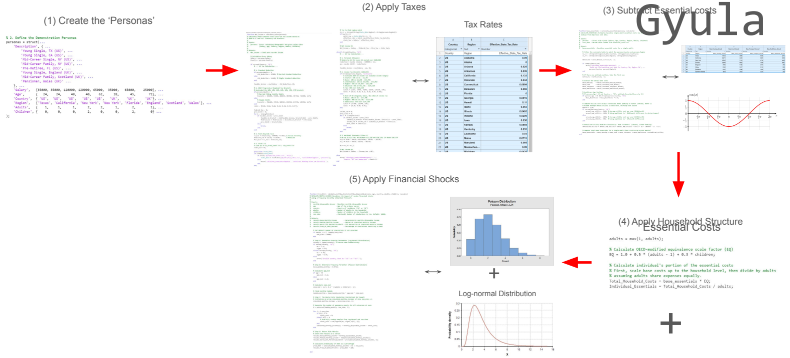
$$s_{i,j} \sim \mathrm{LogNormal}(\mu, \sigma^2), \quad j = 1, 2, \ldots, n_i$$

$$n_i \sim \mathrm{Poisson}(\lambda), \quad i = 1, 2, \ldots, N$$

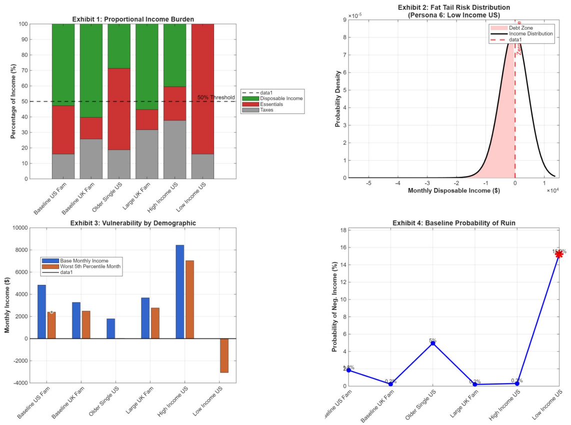
Symbol	Variable
S_m	Shock for Month m
N	Simulation Count
n_i	Shock Count for Simulation i
s	Simulated Shock
μ	Mean for Log of Shock Severity
σ	Std. Deviation for Log of Shock Severity
λ	Mean Shock Rate

Gyula

Model Structure



(6) Acquire and Visualize Results



$$Y_d = \sum_{m=0}^{11} \max\left(Y_t - EQ_{ind}\left(C + V(m)\right) - S_m, 0\right)$$

Symbol	Variable
Y_d	Annual Disposable Income
Y_d	Net Income
EQ_{ind}	OECD-modified Equivalence Scale per person
C	Fixed Monthly Costs
$V(m)$	Variable Monthly Costs for month m
S_m	Financial Shock for month m
m	Month (from 0 to 11)

Gyula

Model Structure

(1) Create the 'Personas'

```
% 2. Define the Demonstration Personas
personas = struct(...
    'Description', { ...
        'Young Single, TX (US)', ...
        'Young Single, CA (US)', ...
        'Mid-Career Single, NY (US)', ...
        'Mid-Career Family, NY (US)', ...
        'Pre-Retiree, FL (US)', ...
        'Young Single, England (UK)', ...
        'Mid-Career Family, Scotland (UK)', ...
        'Pensioner, Wales (UK)' ...
    }, ...
    'Salary',   {35000, 35000, 120000, 120000, 65000,  35000,   65000,    25000}, ...
    'Age',      {   24,    24,     40,     40,    62,     28,      45,       72}, ...
    'Country',  { 'US',  'US',   'US',   'US',  'US',   'UK',    'UK',     'UK'}, ...
    'Region',   {'Texas', 'California', 'New York', 'New York', 'Florida', 'England', 'Scotland', 'Wales'}, ...
    'Adults',   {    1,     1,      1,      2,     2,      1,       2,        1}, ...
    'Children', {    0,     0,      0,      2,     0,      0,       2,        0} ...
);
```

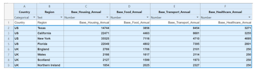
(2) Apply Taxes

Tax Rates

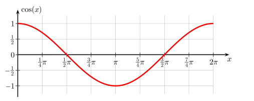
	A	B	C
	Country	Region	Effective_State_Tax_Rate
	Categorical	Text	Number
1	Country	Region	Effective_State_Tax_Rate
2	US	Alabama	0.05
3	US	Alaska	0
4	US	Arizona	0.025
5	US	Arkansas	0.039
6	US	California	0.133
7	US	Colorado	0.044
8	US	Connecticut	0.0699
9	US	Delaware	0.066
10	US	Florida	0
11	US	Georgia	0.0519
12	US	Hawaii	0.11
13	US	Idaho	0.053
14	US	Illinois	0.0495
15	US	Indiana	0.0295
16	US	Iowa	0.038
17	US	Kansas	0.0558
18	US	Kentucky	0.035
19	US	Louisiana	0.03
20	US	Maine	0.0715
21	US	Maryland	0.065
22	US	Massachus...	0.09
23	US	Michigan	0.0425

(3) Subtract Essential costs

Essential Costs



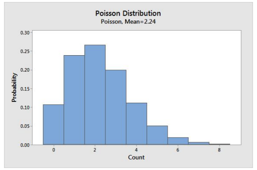
+



(4) Apply Household Structure

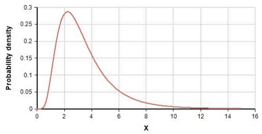
```
adults = max(1, adults);

% Calculate OECD-modified equivalence scale factor (EQ)
EQ = 1.0 + 0.5 * (adults - 1) + 0.3 * children;

% Calculate individual's portion of the essential costs
% First, scale base costs up to the household level, then divide by adults
% assuming adults share expenses equally.
Total_Household_Costs = base_essentials * EQ;
Individual_Essentials = Total_Household_Costs / adults;
```

(5) Apply Financial Shocks

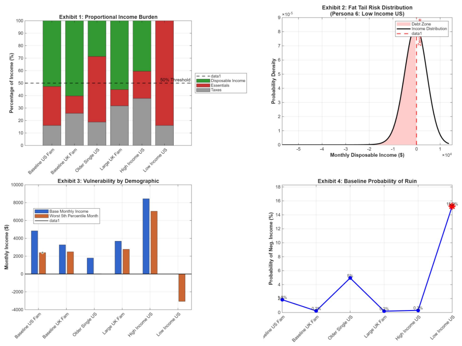


+

Log-normal Distribution



(6) Visualize Results

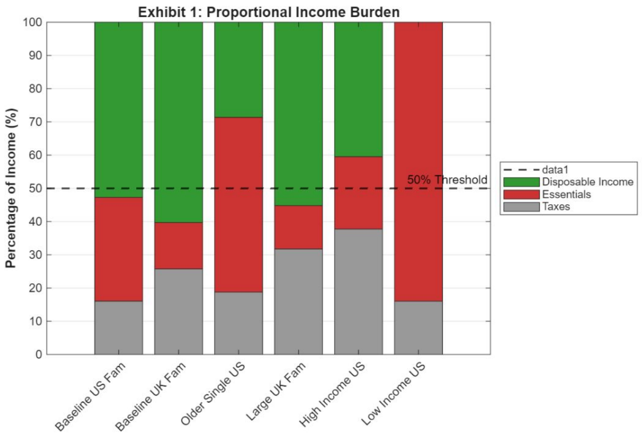


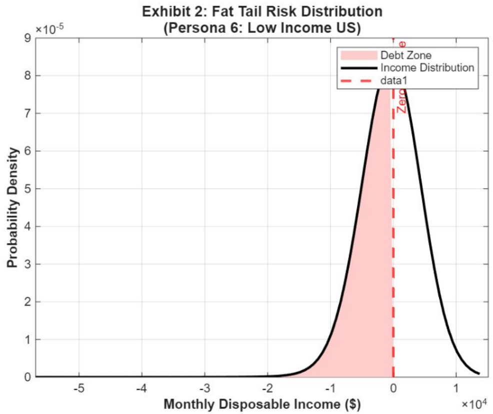
Financial Vulnerability Showcase:

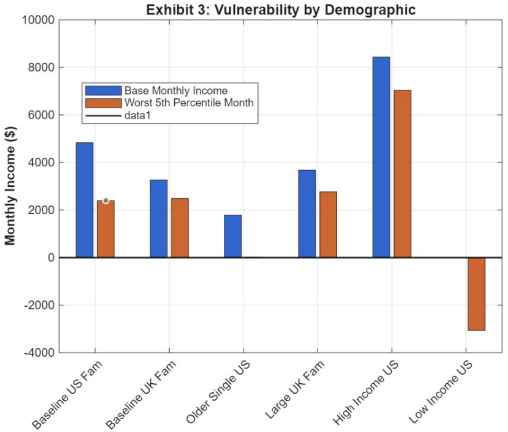
Impact Analysis

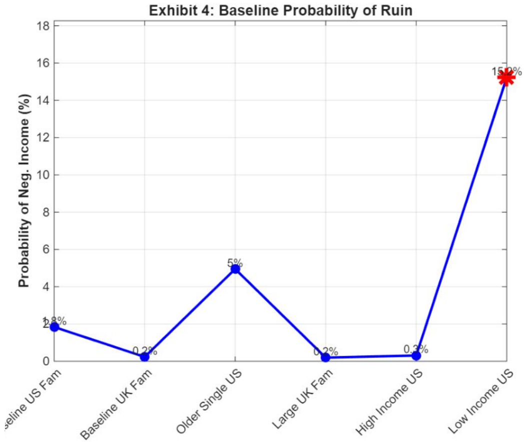
Financial Vulnerability Showcase: Impact Analysis

Stan



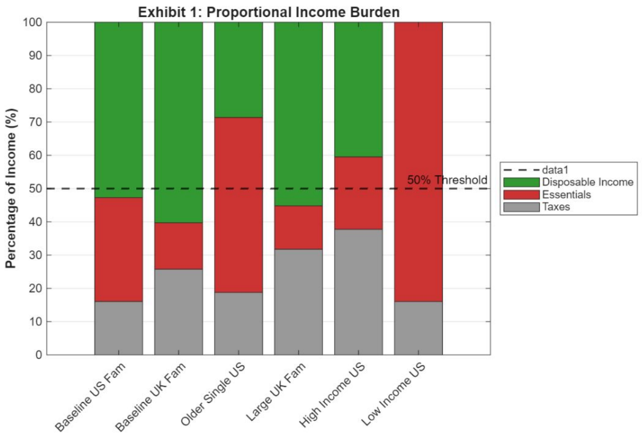


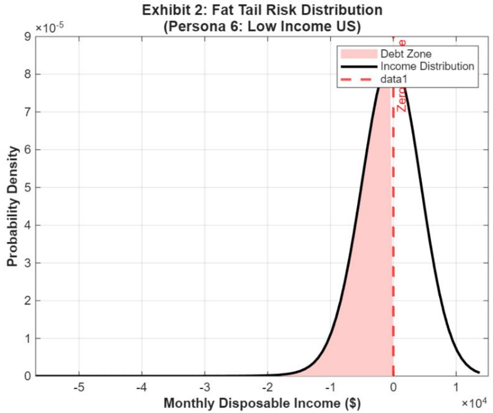


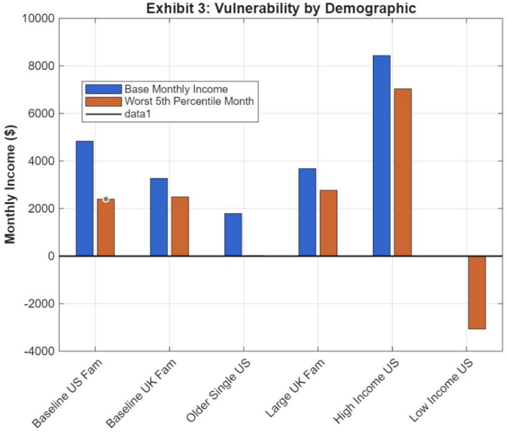


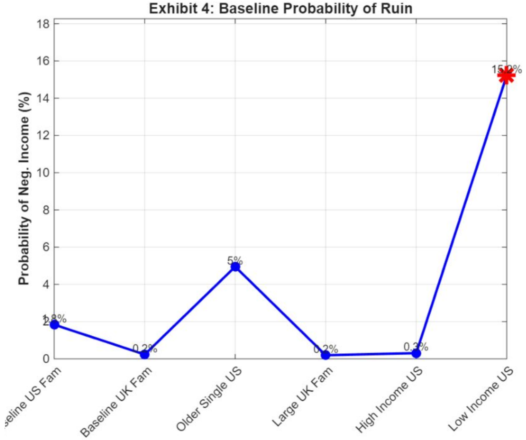
Financial Vulnerability Showcase: Impact Analysis

Stan





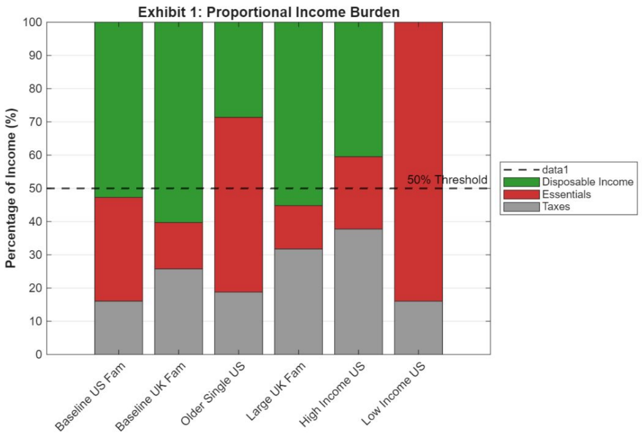


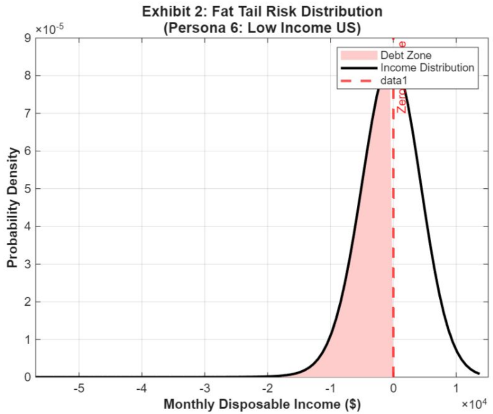


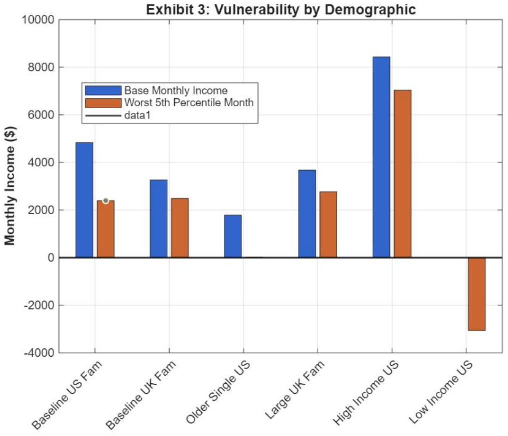
- **Disproportionate Income Burden:** Low-income demographics spend a high percentage of their monthly income on essential needs.

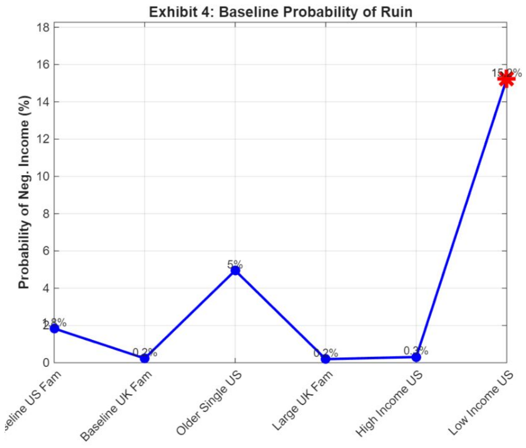
Stan

Financial Vulnerability Showcase: Impact Analysis



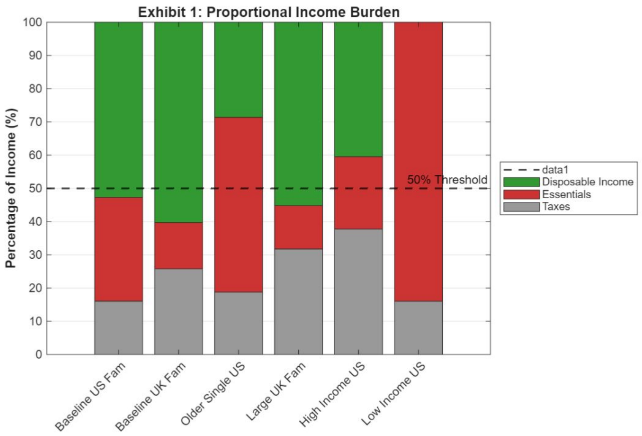


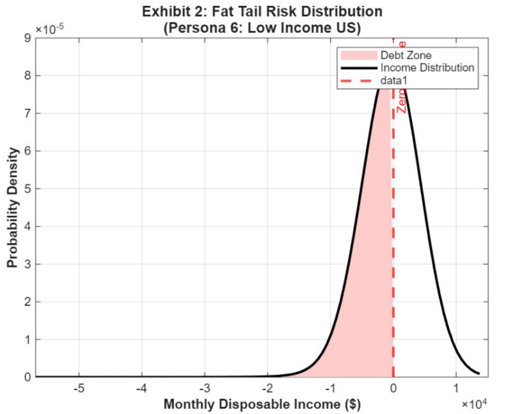


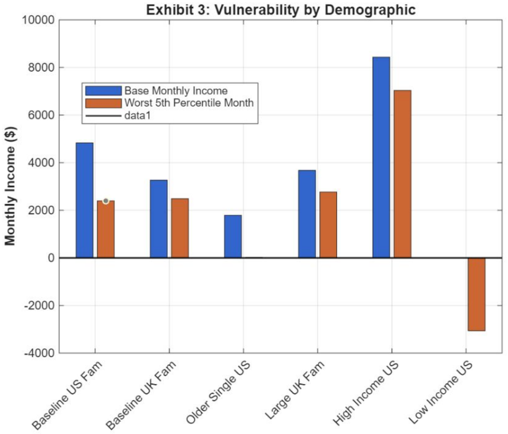


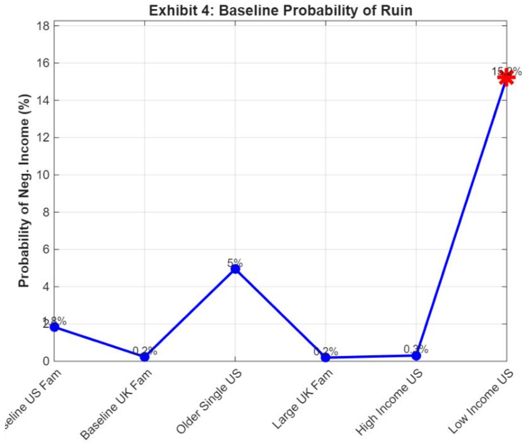
- **Disproportionate Income Burden:** Low-income demographics spend a high percentage of their monthly income on essential needs.
- **Country inequality:** Low to mid income US personas are far more likely to end up in debt than their UK counterparts, whereas high enjoy a much higher disposable income.

Financial Vulnerability Showcase: Impact Analysis





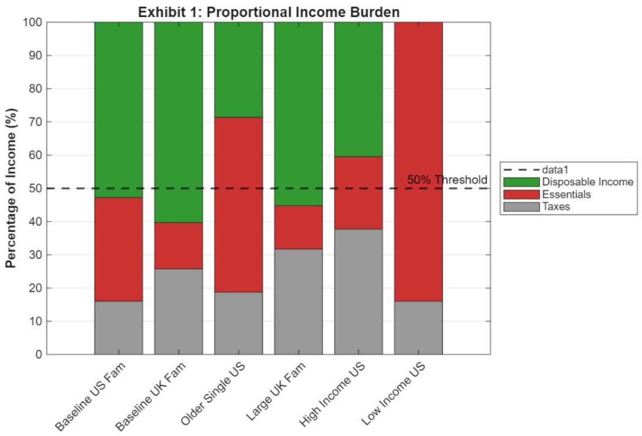


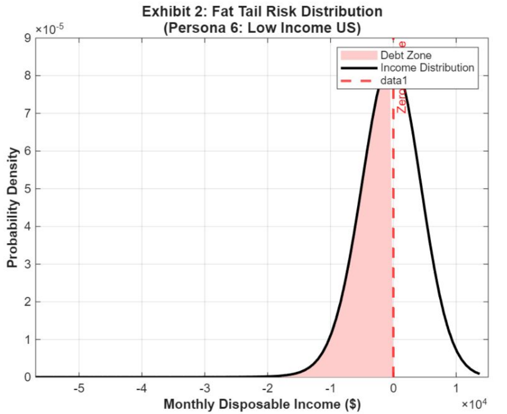


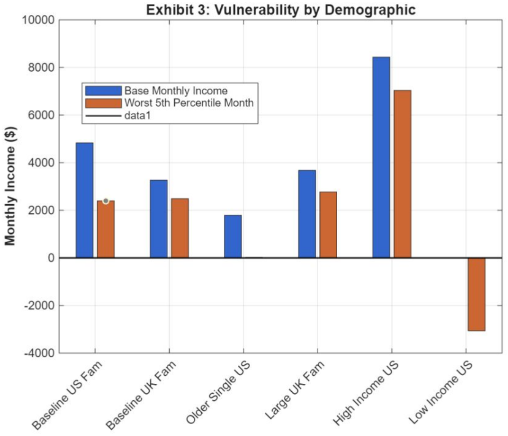
- **Disproportionate Income Burden:** Low-income demographics spend a high percentage of their monthly income on essential needs.
- **Country inequality:** Low to mid income US personas are far more likely to end up in debt than their UK counterparts, whereas high enjoy a much higher disposable income.
- **Severe Vulnerability to Shocks:** When subjected to a 1-in-20 worst-case economic shock, low-income households are driven into severe monthly deficits.

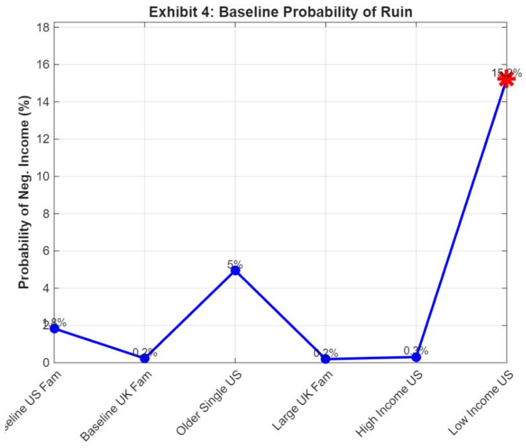
Financial Vulnerability Showcase: Impact Analysis

Stan







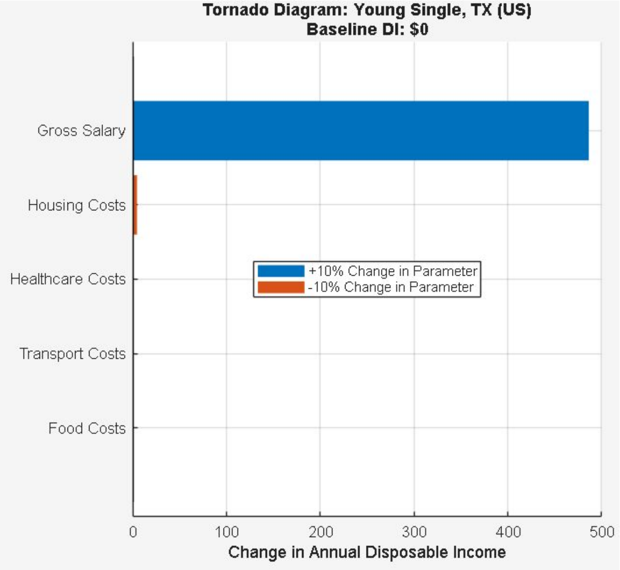


- **Disproportionate Income Burden:** Low-income demographics spend a high percentage of their monthly income on essential needs.
- **Country inequality:** Low to mid income US personas are far more likely to end up in debt than their UK counterparts, whereas high enjoy a much higher disposable income.
- **Severe Vulnerability to Shocks:** When subjected to a 1-in-20 worst-case economic shock, low-income households are driven into severe monthly deficits.
- **Elevated Probability of Debt:** the baseline risk of going into debt simply to cover essential purchases spikes for low-income populations

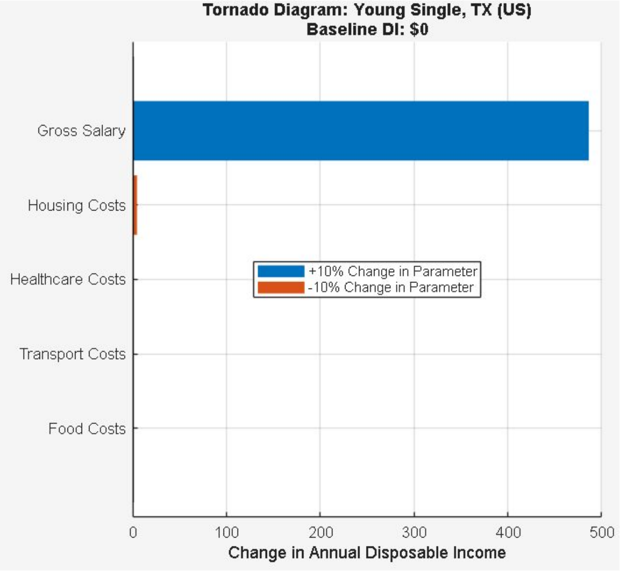
Sensitivity Analysis

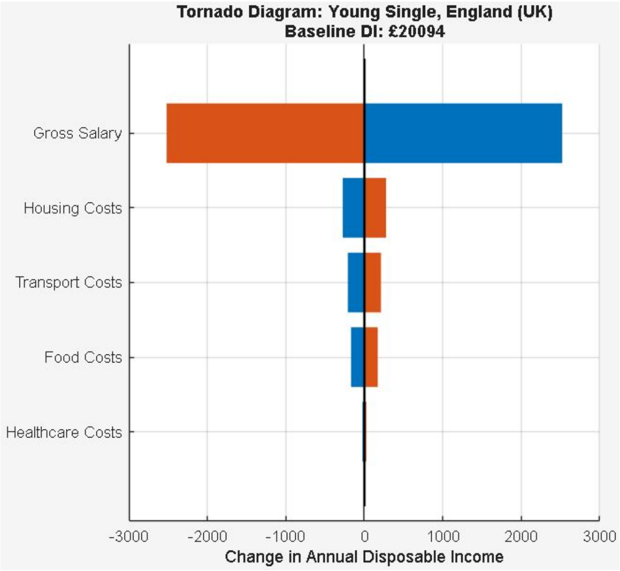
Calvin

Sensitivity Analysis

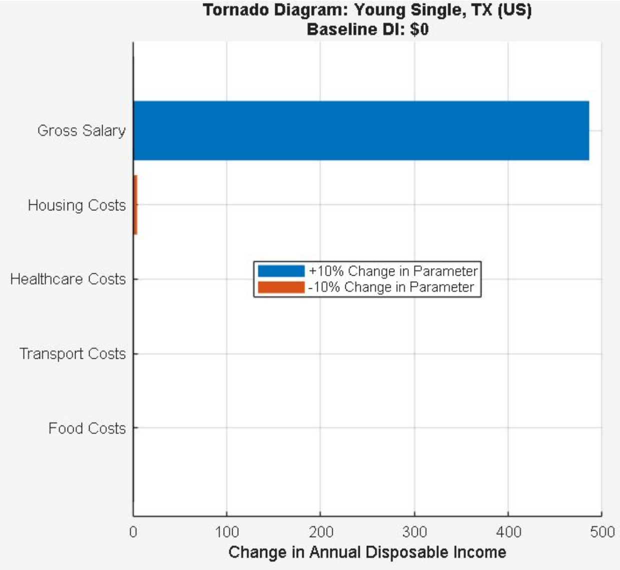


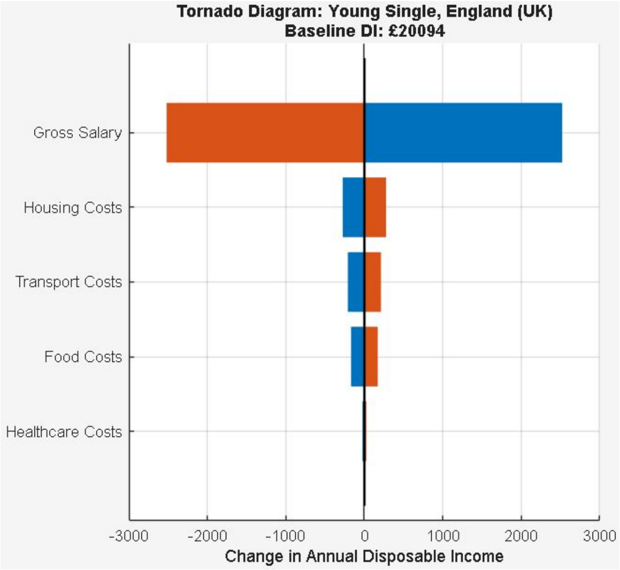
Sensitivity Analysis





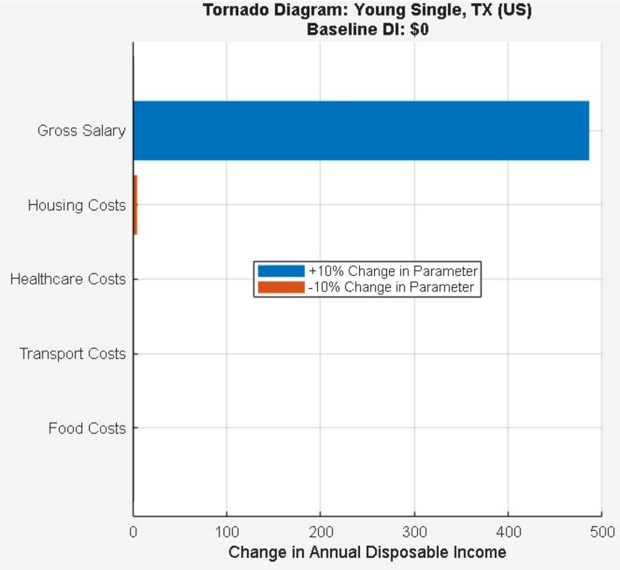
Sensitivity Analysis

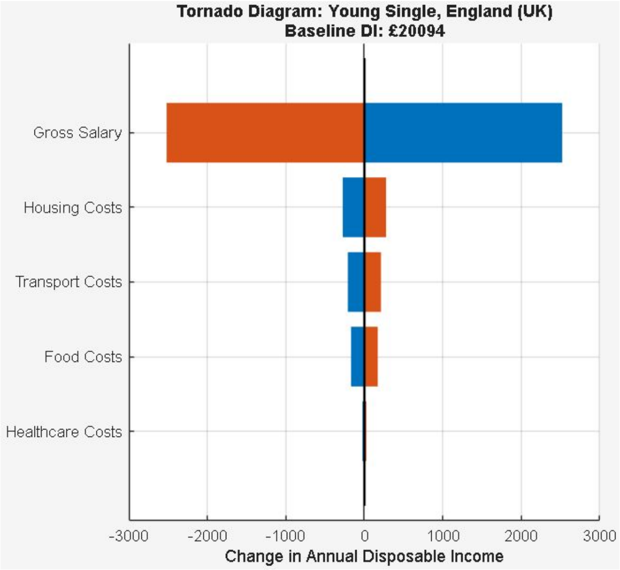




- Local sensitivity analysis → our model is highly stable

Sensitivity Analysis



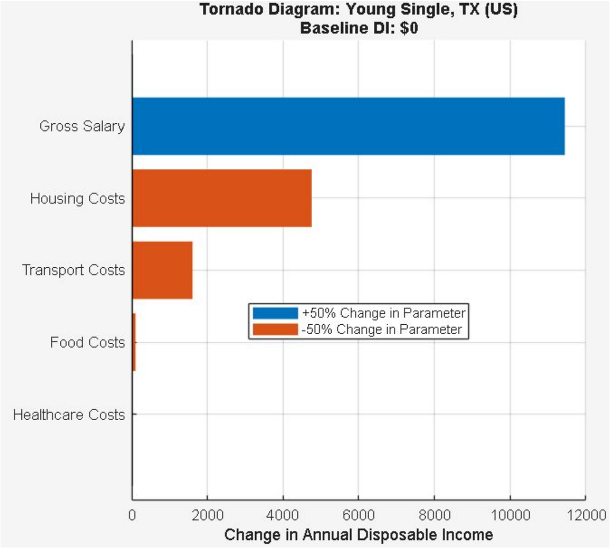


- Local sensitivity analysis → our model is highly stable
  - displaying perfectly symmetric, linear responses to minor parameter changes (e.g., housing or food costs)

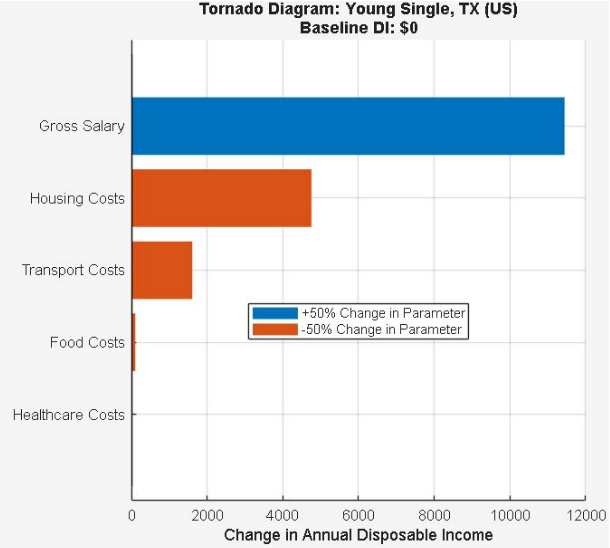
Calvin

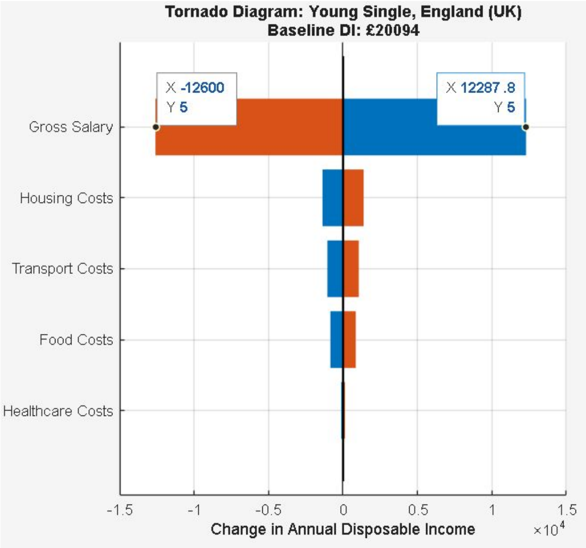
Stress Testing

Stress Testing

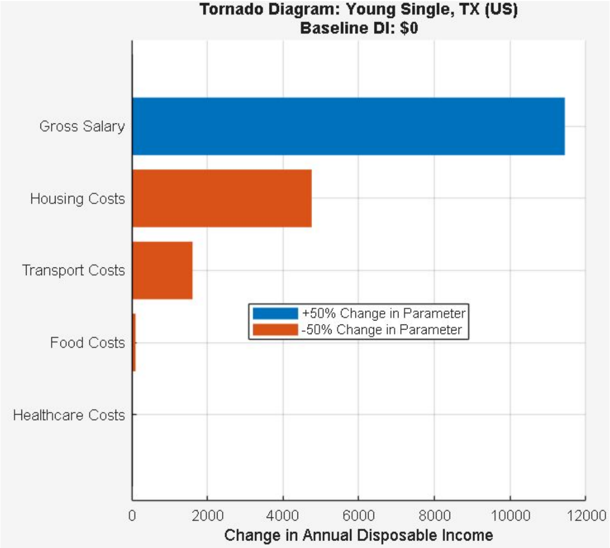


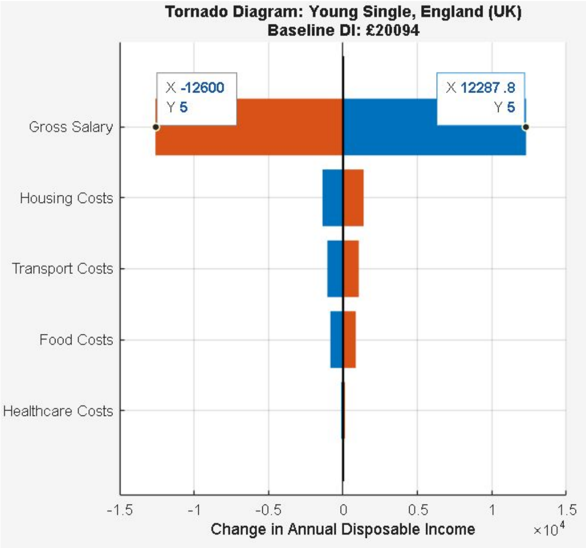
Stress Testing





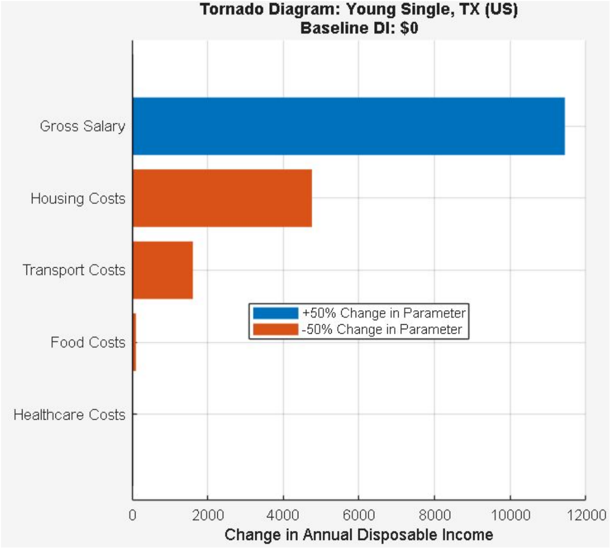
Stress Testing

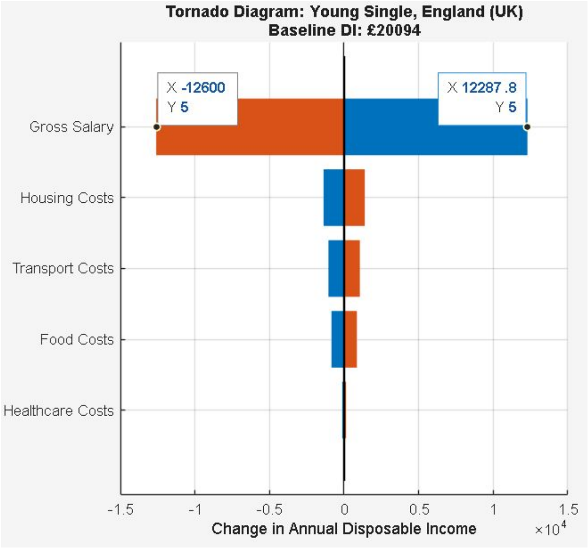




- Stress-testing boundaries, like significant salary increases, breaks this symmetry as personas enter higher tax brackets.

Stress Testing





- Stress-testing boundaries, like significant salary increases, breaks this symmetry as personas enter higher tax brackets.
  - This proves our model isn't a static equation, but structurally adapts to real-world, non-linear financial thresholds.

Model Evaluation

Model Evaluation

- Key strengths

Model Evaluation

- Key strengths
  - Mechanistic & Extensible

Model Evaluation

- Key strengths
  - Mechanistic & Extensible
    - Built on modular logic rather than historical curve-fitting

Model Evaluation

- Key strengths
  - Mechanistic & Extensible
    - Built on modular logic rather than historical curve-fitting
  - High Granularity

Model Evaluation

- Key strengths
  - Mechanistic & Extensible
    - Built on modular logic rather than historical curve-fitting
  - High Granularity
    - Monthly tracking successfully captures seasonal variations, such as winter utility spikes

Model Evaluation

- Key strengths
  - Mechanistic & Extensible
    - Built on modular logic rather than historical curve-fitting
  - High Granularity
    - Monthly tracking successfully captures seasonal variations, such as winter utility spikes
  - Probabilistic Accuracy

Model Evaluation

- Key strengths
  - Mechanistic & Extensible
    - Built on modular logic rather than historical curve-fitting
  - High Granularity
    - Monthly tracking successfully captures seasonal variations, such as winter utility spikes
  - Probabilistic Accuracy
    - Instead of fragile "averages," our Monte Carlo simulation embraces real-world unpredictability

Know the spread

Q2 - an introduction

Q2 - an introduction

❏ Model gambling behaviour evolving over time

Q2 - an introduction

- Model gambling behaviour evolving over time
  - Track risk tolerance, wealth, profit and session behaviour

Q2 - an introduction

- Model gambling behaviour evolving over time
    - Track risk tolerance, wealth, profit and session behaviour
- We built a model with:

Q2 - an introduction

- ❑ Model gambling behaviour evolving over time
  - ❑ Track risk tolerance, wealth, profit and session behaviour
- ❑ We built a model with:
  - ❑ A baseline demographic risk

Q2 - an introduction

- Model gambling behaviour evolving over time
  - Track risk tolerance, wealth, profit and session behaviour
- We built a model with:
  - A baseline demographic risk
  - Dynamic bet selection and stake sizing

Q2 - an introduction

- Model gambling behaviour evolving over time
    - Track risk tolerance, wealth, profit and session behaviour
- We built a model with:
    - A baseline demographic risk
    - Dynamic bet selection and stake sizing
    - Win/loss feedback through loss-chasing

Q2 - an introduction

- Model gambling behaviour evolving over time
  - Track risk tolerance, wealth, profit and session behaviour
- We built a model with:
  - A baseline demographic risk
  - Dynamic bet selection and stake sizing
  - Win/loss feedback through loss-chasing
- This was possible with an agent-based Markov state automaton!

Model Overview

Model Overview

Initialize
Automaton

$$S_{i,m,j,k} = (W_{i,m,j,k},\ \pi_{i,m,j,k},\ \lambda_{i,m,j,k})$$

$$S_{i,1,1,1} = (W_0, 0, \lambda_0)$$

$$S_{i,m,j,1} = (W_{i,m,j}, 0, \lambda_0)$$

$$\lambda_0 = \max(0.01, \min(1, \frac{s}{100}))$$

$$s = \begin{cases} 53.5213 + 1.9007\alpha - 0.0405\alpha^2 & \text{if } g = \text{Male} \\ 54.0832 + 1.5402\alpha - 0.0329\alpha^2 & \text{if } g = \text{Female} \end{cases}$$

Model Overview

Initialize Automaton

$$S_{i,m,j,k} = (W_{i,m,j,k},\ \pi_{i,m,j,k},\ \lambda_{i,m,j,k})$$

$$S_{i,1,1,1} = (W_0, 0, \lambda_0)$$

$$S_{i,m,j,1} = (W_{i,m,j}, 0, \lambda_0)$$

$$\lambda_0 = \max(0.01, \min(1, \frac{s}{100}))$$

$$s = \begin{cases} 53.5213 + 1.9007\alpha - 0.0405\alpha^2 & \text{if } g = \text{Male} \\ 54.0832 + 1.5402\alpha - 0.0329\alpha^2 & \text{if } g = \text{Female} \end{cases}$$

Decide on Bet

$$p = \begin{cases} 0.50 & \text{if } R < 0.08 \\ 0.45 & \text{if } R < 0.15 \\ 0.10 & \text{if } R \geq 0.15 \end{cases} \qquad m = \begin{cases} 0.909 & \text{if } R < 0.08 \\ 1.000 & \text{if } R < 0.15 \\ 7.000 & \text{if } R \geq 0.15 \end{cases}$$

$$R = \lambda_k I$$

$$I = \max(2.0, \min(0.5, \frac{Y}{40000}))$$

Model Overview

Initialize Automaton

$$S_{i,m,j,k} = (W_{i,m,j,k},\ \pi_{i,m,j,k},\ \lambda_{i,m,j,k})$$

$$S_{i,1,1,1} = (W_0, 0, \lambda_0)$$

$$S_{i,m,j,1} = (W_{i,m,j}, 0, \lambda_0)$$

$$\lambda_0 = \max(0.01, \min(1, \frac{s}{100}))$$

$$s = \begin{cases} 53.5213 + 1.9007\alpha - 0.0405\alpha^2 & \text{if } g = \text{Male} \\ 54.0832 + 1.5402\alpha - 0.0329\alpha^2 & \text{if } g = \text{Female} \end{cases}$$

Decide on Bet

$$p = \begin{cases} 0.50 & \text{if } R < 0.08 \\ 0.45 & \text{if } R < 0.15 \\ 0.10 & \text{if } R \geq 0.15 \end{cases} \qquad m = \begin{cases} 0.909 & \text{if } R < 0.08 \\ 1.000 & \text{if } R < 0.15 \\ 7.000 & \text{if } R \geq 0.15 \end{cases}$$

$$R = \lambda_k I$$

$$I = \max(2.0, \min(0.5, \frac{Y}{40000}))$$

Place Bet

$$W_{i,m+1,j,k} = W_{i,m,j,k} + Y$$

$$W_{i,m,j,k+1} = W_{i,m,j,k} + \pi_{i,m,j,k}$$

$$\lambda_{k+1} = \begin{cases} \lambda_0 & \text{if } w_k < p \\ \min(1, c\lambda_k) & \text{if } w_k \geq p \end{cases} \qquad \pi_k = \begin{cases} B_k m & \text{if } w_k < p \\ -B_k & \text{if } w_k \geq p \end{cases}$$

$$w_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n_{i,m,j}$$

$$B_k = \min(B_{max}, B_0 \frac{\lambda_k}{\lambda_0})$$

$$B_0 = 0.001Y \quad B_{max} = 0.05Y$$

Model Overview

Quit or Continue

$$P(S_{j+1,1} \mid S_{j,n_j}) = 1$$

$$P(S_{m+1,1,1} \mid S_{m,B_{freq},n_{m,j}}) = 1$$

$$P(S_{12,B_{freq},n_{m,j}} \mid S_{12,B_{freq},n_{m,j}}) = 1$$

$$S_{k+1} = \begin{cases} S_n & \text{if } W_k + \pi_k \leq 0 \\ S_n & \text{if } \sum_{z=0}^{k} \pi_z < -B_{max} \\ S_n & \text{if } \lambda_k \geq c_k \\ S_{k+1} \neq S_n & \text{otherwise} \end{cases}$$

$$c_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n$$

Initialize Automaton

$$S_{i,m,j,k} = (W_{i,m,j,k},\ \pi_{i,m,j,k},\ \lambda_{i,m,j,k})$$

$$S_{i,1,1,1} = (W_0, 0, \lambda_0)$$

$$S_{i,m,j,1} = (W_{i,m,j}, 0, \lambda_0)$$

$$\lambda_0 = \max(0.01, \min(1, \frac{s}{100}))$$

$$s = \begin{cases} 53.5213 + 1.9007\alpha - 0.0405\alpha^2 & \text{if } g = \text{Male} \\ 54.0832 + 1.5402\alpha - 0.0329\alpha^2 & \text{if } g = \text{Female} \end{cases}$$

Decide on Bet

$$p = \begin{cases} 0.50 & \text{if } R < 0.08 \\ 0.45 & \text{if } R < 0.15 \\ 0.10 & \text{if } R \geq 0.15 \end{cases} \qquad m = \begin{cases} 0.909 & \text{if } R < 0.08 \\ 1.000 & \text{if } R < 0.15 \\ 7.000 & \text{if } R \geq 0.15 \end{cases}$$

$$R = \lambda_k I$$

$$I = \max(2.0, \min(0.5, \frac{Y}{40000}))$$

Place Bet

$$W_{i,m+1,j,k} = W_{i,m,j,k} + Y$$

$$W_{i,m,j,k+1} = W_{i,m,j,k} + \pi_{i,m,j,k}$$

$$\lambda_{k+1} = \begin{cases} \lambda_0 & \text{if } w_k < p \\ \min(1, c\lambda_k) & \text{if } w_k \geq p \end{cases} \qquad \pi_k = \begin{cases} B_k m & \text{if } w_k < p \\ -B_k & \text{if } w_k \geq p \end{cases}$$

$$w_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n_{i,m,j}$$

$$B_k = \min(B_{max}, B_0 \frac{\lambda_k}{\lambda_0})$$

$$B_0 = 0.001Y \quad B_{max} = 0.05Y$$

Model Overview

Initialize Automaton

$$S_{i,m,j,k} = (W_{i,m,j,k},\ \pi_{i,m,j,k},\ \lambda_{i,m,j,k})$$

$$S_{i,1,1,1} = (W_0, 0, \lambda_0)$$

$$S_{i,m,j,1} = (W_{i,m,j}, 0, \lambda_0)$$

$$\lambda_0 = \max(0.01, \min(1, \frac{s}{100}))$$

$$s = \begin{cases} 53.5213 + 1.9007\alpha - 0.0405\alpha^2 & \text{if } g = \text{Male} \\ 54.0832 + 1.5402\alpha - 0.0329\alpha^2 & \text{if } g = \text{Female} \end{cases}$$

Decide on Bet

$$p = \begin{cases} 0.50 & \text{if } R < 0.08 \\ 0.45 & \text{if } R < 0.15 \\ 0.10 & \text{if } R \geq 0.15 \end{cases} \qquad m = \begin{cases} 0.909 & \text{if } R < 0.08 \\ 1.000 & \text{if } R < 0.15 \\ 7.000 & \text{if } R \geq 0.15 \end{cases}$$

$$R = \lambda_k I$$

$$I = \max(2.0, \min(0.5, \frac{Y}{40000}))$$

Place Bet

$$W_{i,m+1,j,k} = W_{i,m,j,k} + Y$$

$$W_{i,m,j,k+1} = W_{i,m,j,k} + \pi_{i,m,j,k}$$

$$\lambda_{k+1} = \begin{cases} \lambda_0 & \text{if } w_k < p \\ \min(1, c\lambda_k) & \text{if } w_k \geq p \end{cases} \qquad \pi_k = \begin{cases} B_k m & \text{if } w_k < p \\ -B_k & \text{if } w_k \geq p \end{cases}$$

$$w_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n_{i,m,j}$$

$$B_k = \min(B_{max}, B_0 \frac{\lambda_k}{\lambda_0})$$

$$B_0 = 0.001Y \quad B_{max} = 0.05Y$$

Quit or Continue

$$P(S_{j+1,1} \mid S_{j,n_j}) = 1$$

$$P(S_{m+1,1,1} \mid S_{m,B_{freq},n_{m,j}}) = 1$$

$$P(S_{12,B_{freq},n_{m,j}} \mid S_{12,B_{freq},n_{m,j}}) = 1$$

$$S_{k+1} = \begin{cases} S_n & \text{if } W_k + \pi_k \leq 0 \\ S_n & \text{if } \sum_{z=0}^{k} \pi_z < -B_{max} \\ S_n & \text{if } \lambda_k \geq c_k \\ S_{k+1} \neq S_n & \text{otherwise} \end{cases}$$

$$c_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n$$

Calculate Total Profits

$$P = \frac{1}{N} \sum_{i=1}^{N} \sum_{m=1}^{12} \sum_{j=1}^{B_{feq}} \sum_{k=1}^{n_{i,m,j}} \pi_{i,m,j,k}$$

Model Overview

Initialize Automaton

$$S_{i,m,j,k} = (W_{i,m,j,k},\ \pi_{i,m,j,k},\ \lambda_{i,m,j,k})$$

$$S_{i,1,1,1} = (W_0, 0, \lambda_0)$$

$$S_{i,m,j,1} = (W_{i,m,j}, 0, \lambda_0)$$

$$\lambda_0 = \max(0.01, \min(1, \frac{s}{100}))$$

$$s = \begin{cases} 53.5213 + 1.9007\alpha - 0.0405\alpha^2 & \text{if } g = \text{Male} \\ 54.0832 + 1.5402\alpha - 0.0329\alpha^2 & \text{if } g = \text{Female} \end{cases}$$

Decide on Bet

$$p = \begin{cases} 0.50 & \text{if } R < 0.08 \\ 0.45 & \text{if } R < 0.15 \\ 0.10 & \text{if } R \geq 0.15 \end{cases} \qquad m = \begin{cases} 0.909 & \text{if } R < 0.08 \\ 1.000 & \text{if } R < 0.15 \\ 7.000 & \text{if } R \geq 0.15 \end{cases}$$

$$R = \lambda_k I$$

$$I = \max(2.0, \min(0.5, \frac{Y}{40000}))$$

Place Bet

$$W_{i,m+1,j,k} = W_{i,m,j,k} + Y$$

$$W_{i,m,j,k+1} = W_{i,m,j,k} + \pi_{i,m,j,k}$$

$$\lambda_{k+1} = \begin{cases} \lambda_0 & \text{if } w_k < p \\ \min(1, c\lambda_k) & \text{if } w_k \geq p \end{cases} \qquad \pi_k = \begin{cases} B_k m & \text{if } w_k < p \\ -B_k & \text{if } w_k \geq p \end{cases}$$

$$w_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n_{i,m,j}$$

$$B_k = \min(B_{max}, B_0 \frac{\lambda_k}{\lambda_0})$$

$$B_0 = 0.001Y \quad B_{max} = 0.05Y$$

Quit or Continue

$$P(S_{j+1,1} \mid S_{j,n_j}) = 1$$

$$P(S_{m+1,1,1} \mid S_{m,B_{freq},n_{m,j}}) = 1$$

$$P(S_{12,B_{freq},n_{m,j}} \mid S_{12,B_{freq},n_{m,j}}) = 1$$

$$S_{k+1} = \begin{cases} S_n & \text{if } W_k + \pi_k \leq 0 \\ S_n & \text{if } \sum_{z=0}^{k} \pi_z < -B_{max} \\ S_n & \text{if } \lambda_k \geq c_k \\ S_{k+1} \neq S_n & \text{otherwise} \end{cases}$$

$$c_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n$$

Calculate Total Profits

$$P = \frac{1}{N} \sum_{i=1}^{N} \sum_{m=1}^{12} \sum_{j=1}^{B_{feq}} \sum_{k=1}^{n_{i,m,j}} \pi_{i,m,j,k}$$

Data used

Data used

- Risk-taking variation with age:

Data used

- Risk-taking variation with age:
  - Journal of Youth and Adolescence

Data used

- Risk-taking variation with age:
    - Journal of Youth and Adolescence
- Risk-taking variation with gender:

Data used

- ❏ Risk-taking variation with age:
  - ❏ Journal of Youth and Adolescence
- ❏ Risk-taking variation with gender:
  - ❏ Gender differences in risk taking: A meta-analysis

Data used

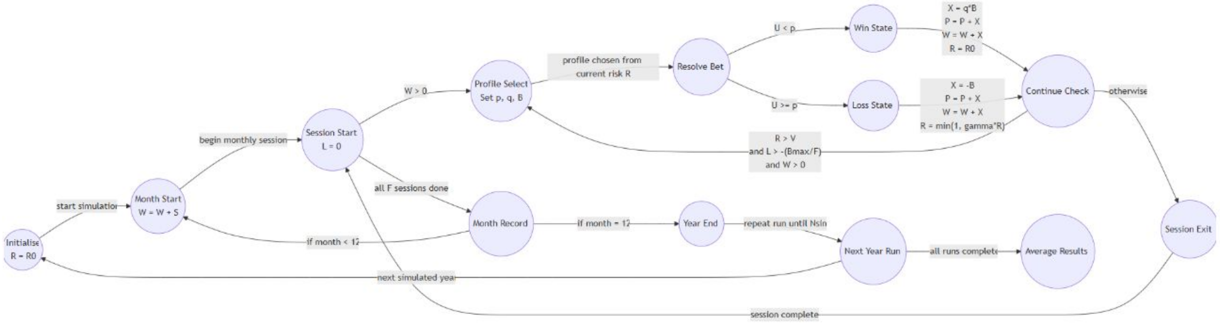
- Risk-taking variation with age:
  - Journal of Youth and Adolescence
- Risk-taking variation with gender:
  - Gender differences in risk taking: A meta-analysis
- Used as base parameters for our simulation

Design overview

Calvin

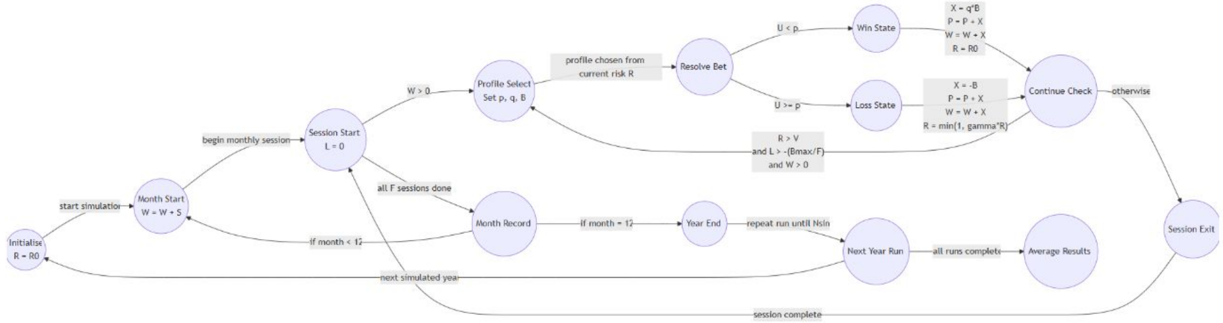
Design overview

1) Markov Automata

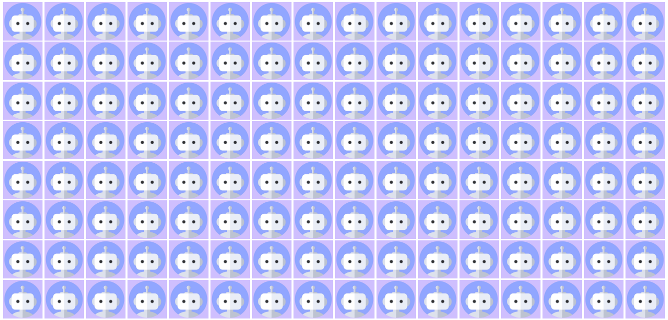


Design overview

1) Markov Automata



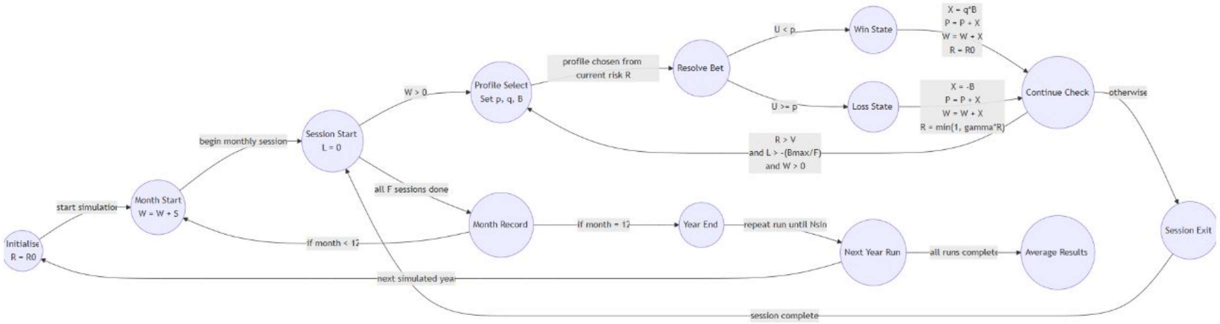
2) Agentic Simulation



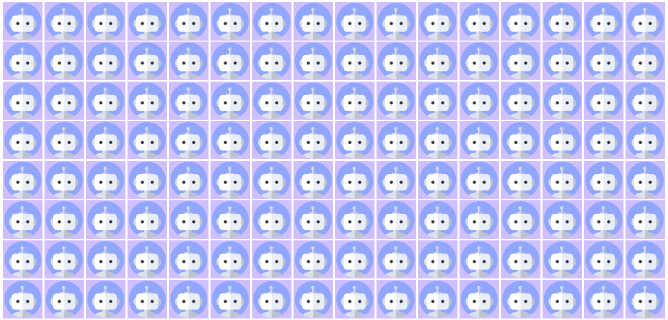
Calvin

Design overview

1) Markov Automata



2) Agentic Simulation



3) Psychological Model

Model Structure

Age Group	Gender	Stoplight Mean	Stoplight S.D.	BART Mean	BART S.D.	Health Risks Mean	Health Risks S.D.	Antisocial Risks Mean	Antisocial Risks S.D.
10—11	Females	41.53	23.57	65.73	11.26	5.15	13.7	14.59	21.59
	Males	41.75	19.72	68.7	11.65	6.71	15.01	22.79	24.77
	Total	41.63	21.79	67.17	11.54	5.9	14.36	18.54	23.53
12—13	Females	39.61	21.54	68.66	11.27	8.9	16.29	18.59	23.25
	Males	43.32	20.18	71.37	11	8.43	14.82	26.74	25.85
	Total	41.53	20.92	70.06	11.2	8.65	15.54	22.79	24.94
14—15	Females	43.37	24.08	70	11.05	16.79	21.98	19.17	22.94
	Males	45.41	21.37	71.93	10.96	15.65	21.34	29.67	27.77
	Total	44.37	22.79	70.96	11.04	16.23	21.66	24.35	25.95
16—17	Females	41.11	22.59	71.58	11.27	24.92	25.2	18.99	21.32
	Males	43.32	20.27	74.2	9.94	30.47	27.02	32.35	26.32
	Total	42.2	21.49	72.88	10.7	27.65	26.24	25.55	24.81
18—19	Females	45.14	23.48	70.68	11.31	30.60	26.63	21.38	21.99
	Males	44.32	20.51	75.70	11.05	41.58	28.81	30.22	26.28
	Total	44.73	22.01	73.25	11.44	36.24	28.27	25.92	24.66
20—21	Females	38.90	22.97	72.36	11.18	34.02	28.36	15.30	18.09
	Males	44.98	22.81	75.52	10.53	47.84	28.56	29.77	25.78
	Total	41.83	23.06	73.89	10.97	40.76	29.25	22.35	23.30
22—23	Females	41.58	23.33	70.89	10.63	34.14	27.08	15.91	19.30
	Males	46.49	21.67	75.85	10.90	49.55	28.63	26.07	24.25
	Total	43.92	22.66	73.27	11.03	41.45	28.84	20.73	22.34
24—25	Females	38.40	23.21	71.34	12.41	38.13	29.01	14.81	17.22
	Males	42.90	22.49	74.27	11.46	52.06	27.50	25.82	23.27
	Total	40.68	22.92	72.82	12.01	45.09	29.07	20.32	21.17
26—27	Females	39.77	21.39	72.18	10.52	41.49	29.25	15.46	21.43
	Males	44.92	24.09	75.53	10.88	52.32	27.63	21.29	22.66
	Total	42.38	22.90	73.86	10.81	46.89	28.92	18.36	22.20
28—30	Females	33.69	22.81	71.68	10.76	34.97	28.69	10.71	17.29
	Males	38.92	23.14	75.41	9.82	51.67	28.90	22.56	23.83
	Total	36.26	23.09	73.46	10.48	42.92	29.94	16.35	21.47

```
% Formula provided: 3.068 + 0.987(Age) - 0.037(Age^2)
score = 54.0832 + 1.5402 * self.Age -0.0329 * (self.Age^2);
% Convert the resulting score to a probability / risk measurement [0, 1]
if self.Gender == "Male"
    score = 53.5213 + 1.9007 * self.Age -0.0405 * (self.Age^2);
end
score = score / 100;
self.BaseRiskTolerance = max(0.01, min(1, score)); % minimum floor of 1%
self.CurrentRiskTolerance = self.BaseRiskTolerance;
```

$$\lambda_0 = \max(0.01, \min(1, \frac{s}{100}))$$

$$s = \begin{cases} 53.5213 + 1.9007\alpha - 0.0405\alpha^2 & \text{if } g = \text{Male} \\ 54.0832 + 1.5402\alpha - 0.0329\alpha^2 & \text{if } g = \text{Female} \end{cases}$$

Symbol	Variable
λ_0	Base Risk Tolerance
s	Estimated BART Score (as percentage)
g	Gender (‘Male’ or ‘Female’)
α	Age

Duell, N. et al. (2018). Age Patterns in Risk Taking Across the World. *Journal of youth and adolescence*, *47*(5), 1052–1072. https://doi.org/10.1007/s10964-017-0752-y

Model Structure

$$S_{i,m,j,k} = (W_{i,m,j,k},\ \pi_{i,m,j,k},\ \lambda_{i,m,j,k})$$

$$S_{i,1,1,1} = (W_0, 0, \lambda_0)$$

$$S_{i,m,j,1} = (W_{i,m,j}, 0, \lambda_0)$$

(1) Markov Automaton Construction - Part 2

```
% Formula provided: 3.068 + 0.987(Age) - 0.037(Age^2)
score = 54.0832 + 1.5402 * self.Age -0.0329 * (self.Age^2);
% Convert the resulting score to a probability / risk measurement [0, 1]
if self.Gender == "Male"
    score = 53.5213 + 1.9007 * self.Age -0.0405 * (self.Age^2);
end
score = score / 100;
self.BaseRiskTolerance = max(0.01, min(1, score)); % minimum floor of 1%
self.CurrentRiskTolerance = self.BaseRiskTolerance;
```

Symbol	Variable
$S_{i,m,j,k}$	Automaton State for agent i, month m, session j, bet m
$W_{i,m,j,k}$	Wealth of Agent for agent i, month m, session j, bet m
W_0	OECD-modified Equivalence Scale per person
λ_0	Base Risk Tolerance
$\pi_{i,m,j,k}$	Profit on Bet for agent i, month m, session j, bet k

Gyula

Model Structure

```
% Formula provided: 3.068 + 0.987(Age) - 0.037(Age^2)
score = 54.0832 + 1.5402 * self.Age -0.0329 * (self.Age^2);
% Convert the resulting score to a probability / risk measurement [0, 1]
if self.Gender == "Male"
    score = 53.5213 + 1.9007 * self.Age -0.0405 * (self.Age^2);
end
score = score / 100;
self.BaseRiskTolerance = max(0.01, min(1, score)); % minimum floor of 1%
self.CurrentRiskTolerance = self.BaseRiskTolerance;
```

$$p = \begin{cases} 0.50 & \text{if } R < 0.08 \\ 0.45 & \text{if } R < 0.15 \\ 0.10 & \text{if } R \geq 0.15 \end{cases} \qquad m = \begin{cases} 0.909 & \text{if } R < 0.08 \\ 1.000 & \text{if } R < 0.15 \\ 7.000 & \text{if } R \geq 0.15 \end{cases}$$

$$R = \lambda_k I$$

$$I = \max(2.0, \min(0.5, \frac{Y}{40000}))$$

(2) Bet Determination

```
function [risk_level, win_prob, payout_multiplier] = determine_bet_profile(self)
    % Incorporate risk tolerance and disposable income into a risk profile
    % Average US disposable income ~40,000 for relative scaling
    income_factor = max(2.0, min(0.5, self.SavedIncome / 40000));
    effective_risk = self.CurrentRiskTolerance * income_factor;

    % Map to defined profiles based on empirical thresholds
    if effective_risk < 0.08
        % Low Risk (Straight Bets)
        risk_level = "Low";
        win_prob = 0.50;
        payout_multiplier = 0.909;
    elseif effective_risk < 0.15
        % Average Risk (Mixed Portfolio)
        risk_level = "Average";
        win_prob = 0.45;
        payout_multiplier = 1.0;
    else
        % High Risk (Parlays)
        risk_level = "High";
        win_prob = 0.10;
        payout_multiplier = 7.0;
    end
end
```

Symbol	Variable
Y	Income per month
I	Income factor for risk-taking
λ_k	Risk Tolerance for for bet k
R	Effective risk Tolerance for bet k
p	Bet Winning Probability
m	Payout Multiplier

Model Structure

Gyula

(1) Markov Automaton Construction

```
% Formula provided: 3.068 + 0.987(Age) - 0.037(Age^2)
score = 54.0832 + 1.5402 * self.Age -0.0329 * (self.Age^2);
% Convert the resulting score to a probability / risk measurement [0, 1]
if self.Gender == "Male"
    score = 53.5213 + 1.9007 * self.Age -0.0405 * (self.Age^2);
end
score = score / 100;
self.BaseRiskTolerance = max(0.01, min(1, score)); % minimum floor of 1%
self.CurrentRiskTolerance = self.BaseRiskTolerance;
```

(2) Bet Determination

```
function [risk_level, win_prob, payout_multiplier] = determine_bet_profile(self)
    % Incorporate risk tolerance and disposable income into a risk profile
    % Average US disposable income ~40,000 for relative scaling
    income_factor = max(2.0, min(0.5, self.SavedIncome / 40000));
    effective_risk = self.CurrentRiskTolerance * income_factor;

    % Map to defined profiles based on empirical thresholds
    if effective_risk < 0.08
        % Low Risk (Straight Bets)
        risk_level = "Low";
        win_prob = 0.50;
        payout_multiplier = 0.909;
    elseif effective_risk < 0.15
        % Average Risk (Mixed Portfolio)
        risk_level = "Average";
        win_prob = 0.45;
        payout_multiplier = 1.0;
    else
        % High Risk (Parlays)
        risk_level = "High";
        win_prob = 0.10;
        payout_multiplier = 7.0;
    end
end
```

(3) Placing the Bet - Part 1

```
function profit = place_bet(self, bet_amount)
    [~, win_prob, payout_multiplier] = self.determine_bet_profile();

    % Simulate outcome
    is_win = rand() < win_prob;

    if is_win
        % Gambler wins
        profit = bet_amount * payout_multiplier;
        self.NetProfit = self.NetProfit + profit;

        % Reset risk tolerance back to original value (Markov Chain restart)
        self.CurrentRiskTolerance = self.BaseRiskTolerance;
    else
        % Gambler loses
        profit = -bet_amount;
        self.NetProfit = self.NetProfit + profit;

        % Chase losses: multiply risk tolerance by a chasing coefficient > 1
        self.CurrentRiskTolerance = min(1.0, self.CurrentRiskTolerance * self.ChasingCoefficient);
    end
end
```

$$w_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n_{i,m,j}$$

$$B_k = \min(B_{max}, B_0 \frac{\lambda_k}{\lambda_0})$$

$$B_0 = 0.001Y \quad B_{max} = 0.05Y$$

Symbol	Variable
w_k	Winning Boundary
B_k	Betting amount for bet k
B_{max}	Maximum Betting Amount
B_0	Minimum Betting Amount
λ_k	Risk Tolerance for bet k
λ_0	Base Risk Tolerance

Model Structure

Gyula

(1) Markov Automaton Construction

```
% Formula provided: 3.068 + 0.987(Age) - 0.037(Age^2)
score = 54.0832 + 1.5402 * self.Age -0.0329 * (self.Age^2);
% Convert the resulting score to a probability / risk measurement [0, 1]
if self.Gender == "Male"
    score = 53.5213 + 1.9007 * self.Age -0.0405 * (self.Age^2);
end
score = score / 100;
self.BaseRiskTolerance = max(0.01, min(1, score)); % minimum floor of 1%
self.CurrentRiskTolerance = self.BaseRiskTolerance;
```

(2) Bet Determination

```
function [risk_level, win_prob, payout_multiplier] = determine_bet_profile(self)
    % Incorporate risk tolerance and disposable income into a risk profile
    % Average US disposable income ~40,000 for relative scaling
    income_factor = max(2.0, min(0.5, self.SavedIncome / 40000));
    effective_risk = self.CurrentRiskTolerance * income_factor;

    % Map to defined profiles based on empirical thresholds
    if effective_risk < 0.08
        % Low Risk (Straight Bets)
        risk_level = "Low";
        win_prob = 0.50;
        payout_multiplier = 0.909;
    elseif effective_risk < 0.15
        % Average Risk (Mixed Portfolio)
        risk_level = "Average";
        win_prob = 0.45;
        payout_multiplier = 1.0;
    else
        % High Risk (Parlays)
        risk_level = "High";
        win_prob = 0.10;
        payout_multiplier = 7.0;
    end
end
```

(3) Placing the Bet - Part 2

$$\lambda_{k+1} = \begin{cases} \lambda_0 & \text{if } w_k < p \\ \min(1, c\lambda_k) & \text{if } w_k \geq p \end{cases} \qquad \pi_k = \begin{cases} B_k m & \text{if } w_k < p \\ -B_k & \text{if } w_k \geq p \end{cases}$$

```
function profit = place_bet(self, bet_amount)
    [~, win_prob, payout_multiplier] = self.determine_bet_profile();

    % Simulate outcome
    is_win = rand() < win_prob;

    if is_win
        % Gambler wins
        profit = bet_amount * payout_multiplier;
        self.NetProfit = self.NetProfit + profit;

        % Reset risk tolerance back to original value (Markov Chain restart)
        self.CurrentRiskTolerance = self.BaseRiskTolerance;
    else
        % Gambler loses
        profit = -bet_amount;
        self.NetProfit = self.NetProfit + profit;

        % Chase losses: multiply risk tolerance by a chasing coefficient > 1
        self.CurrentRiskTolerance = min(1.0, self.CurrentRiskTolerance * self.ChasingCoefficient);
    end
end
```

Symbol	Variable
λ_k	Risk Tolerance for bet *k*
λ_0	Base Risk Tolerance
c	Chasing coef.
w_k	Winning Boundary
p	Probability of winning
π_k	Profit for bet *k*
B_k	Bet Amount for bet *k*
m	Payout multiplier

Model Structure

Gyula

(1) Markov Automaton Construction

```
% Formula provided: 3.068 + 0.987(Age) - 0.037(Age^2)
score = 54.0832 + 1.5402 * self.Age -0.0329 * (self.Age^2);
% Convert the resulting score to a probability / risk measurement [0, 1]
if self.Gender == "Male"
    score = 53.5213 + 1.9007 * self.Age -0.0405 * (self.Age^2);
end
score = score / 100;
self.BaseRiskTolerance = max(0.01, min(1, score)); % minimum floor of 1%
self.CurrentRiskTolerance = self.BaseRiskTolerance;
```

(2) Bet Determination

```
function [risk_level, win_prob, payout_multiplier] = determine_bet_profile(self)
    % Incorporate risk tolerance and disposable income into a risk profile
    % Average US disposable income ~40,000 for relative scaling
    income_factor = max(2.0, min(0.5, self.SavedIncome / 40000));
    effective_risk = self.CurrentRiskTolerance * income_factor;

    % Map to defined profiles based on empirical thresholds
    if effective_risk < 0.08
        % Low Risk (Straight Bets)
        risk_level = "Low";
        win_prob = 0.50;
        payout_multiplier = 0.909;
    elseif effective_risk < 0.15
        % Average Risk (Mixed Portfolio)
        risk_level = "Average";
        win_prob = 0.45;
        payout_multiplier = 1.0;
    else
        % High Risk (Parlays)
        risk_level = "High";
        win_prob = 0.10;
        payout_multiplier = 7.0;
    end
end
```

(3) Placing the Bet - Part 3

```
function profit = place_bet(self, bet_amount)
    [~, win_prob, payout_multiplier] = self.determine_bet_profile();

    % Simulate outcome
    is_win = rand() < win_prob;

    if is_win
        % Gambler wins
        profit = bet_amount * payout_multiplier;
        self.NetProfit = self.NetProfit + profit;

        % Reset risk tolerance back to original value (Markov Chain restart)
        self.CurrentRiskTolerance = self.BaseRiskTolerance;
    else
        % Gambler loses
        profit = -bet_amount;
        self.NetProfit = self.NetProfit + profit;

        % Chase losses: multiply risk tolerance by a chasing coefficient > 1
        self.CurrentRiskTolerance = min(1.0, self.CurrentRiskTolerance * self.ChasingCoefficient);
    end
end
```

$$W_{i,m+1,j,k} = W_{i,m,j,k} + Y$$

$$W_{i,m,j,k+1} = W_{i,m,j,k} + \pi_{i,m,j,k}$$

Symbol	Variable
$W_{i,m,j,k}$	Agent's wealth for agent i, month m, session j, bet k
Y	Monthly income
$\pi_{i,m,j,k}$	Profit on Bet for agent i, month m, session j, bet k

Model Structure

Gyula

(1) Markov Automaton Construction

```
% Formula provided: 3.068 + 0.987(Age) - 0.037(Age^2)
score = 54.0832 + 1.5402 * self.Age -0.0329 * (self.Age^2);
% Convert the resulting score to a probability / risk measurement [0, 1]
if self.Gender == "Male"
    score = 53.5213 + 1.9007 * self.Age -0.0405 * (self.Age^2);
end
score = score / 100;
self.BaseRiskTolerance = max(0.01, min(1, score)); % minimum floor of 1%
self.CurrentRiskTolerance = self.BaseRiskTolerance;
```

(2) Bet Determination

```
function [risk_level, win_prob, payout_multiplier] = determine_bet_profile(self)
    % Incorporate risk tolerance and disposable income into a risk profile
    % Average US disposable income ~40,000 for relative scaling
    income_factor = max(2.0, min(0.5, self.SavedIncome / 40000));
    effective_risk = self.CurrentRiskTolerance * income_factor;

    % Map to defined profiles based on empirical thresholds
    if effective_risk < 0.08
        % Low Risk (Straight Bets)
        risk_level = "Low";
        win_prob = 0.50;
        payout_multiplier = 0.909;
    elseif effective_risk < 0.15
        % Average Risk (Mixed Portfolio)
        risk_level = "Average";
        win_prob = 0.45;
        payout_multiplier = 1.0;
    else
        % High Risk (Parlays)
        risk_level = "High";
        win_prob = 0.10;
        payout_multiplier = 7.0;
    end
end
```

(3) Placing the Bet

```
function profit = place_bet(self, bet_amount)
    [~, win_prob, payout_multiplier] = self.determine_bet_profile();

    % Simulate outcome
    is_win = rand() < win_prob;

    if is_win
        % Gambler wins
        profit = bet_amount * payout_multiplier;
        self.NetProfit = self.NetProfit + profit;

        % Reset risk tolerance back to original value (Markov Chain restart)
        self.CurrentRiskTolerance = self.BaseRiskTolerance;
    else
        % Gambler loses
        profit = -bet_amount;
        self.NetProfit = self.NetProfit + profit;

        % Chase losses: multiply risk tolerance by a chasing coefficient > 1
        self.CurrentRiskTolerance = min(1.0, self.CurrentRiskTolerance * self.ChasingCoefficient);
    end
end
```

(4) Automaton Termination - Part 1

```
function simulate_year(self, bettingFreq)
    self.BetHistory = zeros(12, 1);
    self.ProfitHistory = zeros(12, 1);

    for i = 1:12
        self.Wealth = self.SavedIncome + self.Wealth;

        for f = 1:bettingFreq
            total_profit = 0;
            betting = self.Wealth > 0;
            base_bet_amount = 0.001 * self.SavedIncome;
            max_bet = self.SavedIncome * 0.05;

            while betting
                bet_amount = base_bet_amount * (self.CurrentRiskTolerance / self.BaseRiskTolerance);
                bet_amount = min(bet_amount, max_bet);
                outcome = self.place_bet(bet_amount);
                total_profit = total_profit + outcome;
                self.Wealth = self.Wealth + outcome;
                out = rand();
                cond = true;
                if self.max_bet_per_month
                    cond = total_profit > (-max_bet);
                else
                    cond = total_profit > (-max_bet/bettingFreq);
                end

                betting = self.CurrentRiskTolerance > out && cond && self.Wealth > 0;
            end

            self.NetProfit = self.NetProfit + total_profit;
        end
        % Bet size scales with risk tolerance, capped at 5% of disposable income


        self.ProfitHistory(i) = self.NetProfit;
    end
end
```

$$S_{k+1} = \begin{cases} S_n & \text{if } W_k + \pi_k \leq 0 \\ S_n & \text{if } \sum_{z=0}^{k} \pi_z < -B_{max} \\ S_n & \text{if } \lambda_k \geq c_k \\ S_{k+1} \neq S_n & \text{otherwise} \end{cases}$$

$$c_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n$$

Symbol	Variable
S_{k+1}	Automaton State after bet k
S_n	Final Automaton State for a Session
W_k	Wealth before bet k
π_k	Profit on bet k
B_{max}	Maximum Bet Amount
λ_k	Risk Tolerance for bet k
c_k	Stochastic Termination Boundary

Model Structure

Gyula

(1) Markov Automaton Construction

```
% Formula provided: 3.068 + 0.987(Age) - 0.037(Age^2)
score = 54.0832 + 1.5402 * self.Age -0.0329 * (self.Age^2);
% Convert the resulting score to a probability / risk measurement [0, 1]
if self.Gender == "Male"
    score = 53.5213 + 1.9007 * self.Age -0.0405 * (self.Age^2);
end
score = score / 100;
self.BaseRiskTolerance = max(0.01, min(1, score)); % minimum floor of 1%
self.CurrentRiskTolerance = self.BaseRiskTolerance;
```

(2) Bet Determination

```
function [risk_level, win_prob, payout_multiplier] = determine_bet_profile(self)
    % Incorporate risk tolerance and disposable income into a risk profile
    % Average US disposable income ~40,000 for relative scaling
    income_factor = max(2.0, min(0.5, self.SavedIncome / 40000));
    effective_risk = self.CurrentRiskTolerance * income_factor;

    % Map to defined profiles based on empirical thresholds
    if effective_risk < 0.08
        % Low Risk (Straight Bets)
        risk_level = "Low";
        win_prob = 0.50;
        payout_multiplier = 0.909;
    elseif effective_risk < 0.15
        % Average Risk (Mixed Portfolio)
        risk_level = "Average";
        win_prob = 0.45;
        payout_multiplier = 1.0;
    else
        % High Risk (Parlays)
        risk_level = "High";
        win_prob = 0.10;
        payout_multiplier = 7.0;
    end
end
```

(3) Placing the Bet

```
function profit = place_bet(self, bet_amount)
    [~, win_prob, payout_multiplier] = self.determine_bet_profile();

    % Simulate outcome
    is_win = rand() < win_prob;

    if is_win
        % Gambler wins
        profit = bet_amount * payout_multiplier;
        self.NetProfit = self.NetProfit + profit;

        % Reset risk tolerance back to original value (Markov Chain restart)
        self.CurrentRiskTolerance = self.BaseRiskTolerance;
    else
        % Gambler loses
        profit = -bet_amount;
        self.NetProfit = self.NetProfit + profit;

        % Chase losses: multiply risk tolerance by a chasing coefficient > 1
        self.CurrentRiskTolerance = min(1.0, self.CurrentRiskTolerance * self.ChasingCoefficient);
    end
end
```

(4) Automaton Termination - Part 2

```
function simulate_year(self, bettingFreq)
    self.BetHistory = zeros(12, 1);
    self.ProfitHistory = zeros(12, 1);

    for i = 1:12
        self.Wealth = self.SavedIncome + self.Wealth;

        for f = 1:bettingFreq
            total_profit = 0;
            betting = self.Wealth > 0;
            base_bet_amount = 0.001 * self.SavedIncome;
            max_bet = self.SavedIncome * 0.05;

            while betting
                bet_amount = base_bet_amount * (self.CurrentRiskTolerance / self.BaseRiskTolerance);
                bet_amount = min(bet_amount, max_bet);
                outcome = self.place_bet(bet_amount);
                total_profit = total_profit + outcome;
                self.Wealth = self.Wealth + outcome;
                out = rand();
                cond = true;
                if self.max_bet_per_month
                    cond = total_profit > (-max_bet);
                else
                    cond = total_profit > (-max_bet/bettingFreq);
                end

                betting = self.CurrentRiskTolerance > out && cond && self.Wealth > 0;
            end

            self.NetProfit = self.NetProfit + total_profit;
        end
        % Bet size scales with risk tolerance, capped at 5% of disposable income


        self.ProfitHistory(i) = self.NetProfit;
    end
end
```

$$P(S_{j+1,1} \mid S_{j,n_j}) = 1$$

$$P(S_{m+1,1,1} \mid S_{m,B_{freq},n_{m,j}}) = 1$$

$$P(S_{12,B_{freq},n_{m,j}} \mid S_{12,B_{freq},n_{m,j}}) = 1$$

Symbol	Variable
$S_{m,j,k}$	Agent’s State for month m, session j, bet k
B_{freq}	Number of Betting Sessions per Month
$n_{m,j}$	Number of bets for month m, session j

Model Structure

Gyula

(1) Markov Automaton Construction

```
% Formula provided: 3.068 + 0.987(Age) - 0.037(Age^2)
score = 54.0832 + 1.5402 * self.Age -0.0329 * (self.Age^2);
% Convert the resulting score to a probability / risk measurement [0, 1]
if self.Gender == "Male"
    score = 53.5213 + 1.9007 * self.Age -0.0405 * (self.Age^2);
end
score = score / 100;
self.BaseRiskTolerance = max(0.01, min(1, score)); % minimum floor of 1%
self.CurrentRiskTolerance = self.BaseRiskTolerance;
```

(2) Bet Determination

```
function [risk_level, win_prob, payout_multiplier] = determine_bet_profile(self)
    % Incorporate risk tolerance and disposable income into a risk profile
    % Average US disposable income ~40,000 for relative scaling
    income_factor = max(2.0, min(0.5, self.SavedIncome / 40000));
    effective_risk = self.CurrentRiskTolerance * income_factor;

    % Map to defined profiles based on empirical thresholds
    if effective_risk < 0.08
        % Low Risk (Straight Bets)
        risk_level = "Low";
        win_prob = 0.50;
        payout_multiplier = 0.909;
    elseif effective_risk < 0.15
        % Average Risk (Mixed Portfolio)
        risk_level = "Average";
        win_prob = 0.45;
        payout_multiplier = 1.0;
    else
        % High Risk (Parlays)
        risk_level = "High";
        win_prob = 0.10;
        payout_multiplier = 7.0;
    end
end
```

(3) Placing the Bet

```
function profit = place_bet(self, bet_amount)
    [~, win_prob, payout_multiplier] = self.determine_bet_profile();

    % Simulate outcome
    is_win = rand() < win_prob;

    if is_win
        % Gambler wins
        profit = bet_amount * payout_multiplier;
        self.NetProfit = self.NetProfit + profit;

        % Reset risk tolerance back to original value (Markov Chain restart)
        self.CurrentRiskTolerance = self.BaseRiskTolerance;
    else
        % Gambler loses
        profit = -bet_amount;
        self.NetProfit = self.NetProfit + profit;

        % Chase losses: multiply risk tolerance by a chasing coefficient > 1
        self.CurrentRiskTolerance = min(1.0, self.CurrentRiskTolerance * self.ChasingCoefficient);
    end
end
```

(4) Automaton Termination

```
function simulate_year(self, bettingFreq)
    self.BetHistory = zeros(12, 1);
    self.ProfitHistory = zeros(12, 1);

    for i = 1:12
        self.Wealth = self.SavedIncome + self.Wealth;

        for f = 1:bettingFreq
            total_profit = 0;
            betting = self.Wealth > 0;
            base_bet_amount = 0.001 * self.SavedIncome;
            max_bet = self.SavedIncome * 0.05;

            while betting
                bet_amount = base_bet_amount * (self.CurrentRiskTolerance / self.BaseRiskTolerance);
                bet_amount = min(bet_amount, max_bet);
                outcome = self.place_bet(bet_amount);
                total_profit = total_profit + outcome;
                self.Wealth = self.Wealth + outcome;
                out = rand();
                cond = true;
                if self.max_bet_per_month
                    cond = total_profit > (-max_bet);
                else
                    cond = total_profit > (-max_bet/bettingFreq);
                end

                betting = self.CurrentRiskTolerance > out && cond && self.Wealth > 0;
            end

            self.NetProfit = self.NetProfit + total_profit;
        end
        % Bet size scales with risk tolerance, capped at 5% of disposable income


        self.ProfitHistory(i) = self.NetProfit;
    end
end
```

$$P = \frac{1}{N} \sum_{i=1}^{N} \sum_{m=1}^{12} \sum_{j=1}^{B_{feq}} \sum_{k=1}^{n_{i,m,j}} \pi_{i,m,j,k}$$

Symbol	Variable
N	Number of Agents/Simulations
B_{freq}	Number of Betting Sessions per Month
$n_{i,m,j}$	Number of bets for agent *i*, month *m*, session *j*
$\pi_{i,m,j,k}$	Profit on Bet for agent *i*, month *m*, session *j*, bet *k*

Gyula

Model Overview

Initialize Automaton

$$S_{i,m,j,k} = (W_{i,m,j,k},\ \pi_{i,m,j,k},\ \lambda_{i,m,j,k})$$

$$S_{i,1,1,1} = (W_0, 0, \lambda_0)$$

$$S_{i,m,j,1} = (W_{i,m,j}, 0, \lambda_0)$$

$$\lambda_0 = \max(0.01, \min(1, \frac{s}{100}))$$

$$s = \begin{cases} 53.5213 + 1.9007\alpha - 0.0405\alpha^2 & \text{if } g = \text{Male} \\ 54.0832 + 1.5402\alpha - 0.0329\alpha^2 & \text{if } g = \text{Female} \end{cases}$$

Quit or Continue

$$P(S_{j+1,1} \mid S_{j,n_j}) = 1$$

$$P(S_{m+1,1,1} \mid S_{m,B_{freq},n_{m,j}}) = 1$$

$$P(S_{12,B_{freq},n_{m,j}} \mid S_{12,B_{freq},n_{m,j}}) = 1$$

$$S_{k+1} = \begin{cases} S_n & \text{if } W_k + \pi_k \le 0 \\ S_n & \text{if } \sum_{z=0}^{k} \pi_z < -B_{max} \\ S_n & \text{if } \lambda_k \ge c_k \\ S_{k+1} \ne S_n & \text{otherwise} \end{cases}$$

$$c_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n$$

Decide on Bet

$$p = \begin{cases} 0.50 & \text{if } R < 0.08 \\ 0.45 & \text{if } R < 0.15 \\ 0.10 & \text{if } R \ge 0.15 \end{cases} \qquad m = \begin{cases} 0.909 & \text{if } R < 0.08 \\ 1.000 & \text{if } R < 0.15 \\ 7.000 & \text{if } R \ge 0.15 \end{cases}$$

$$R = \lambda_k I$$

$$I = \max(2.0, \min(0.5, \frac{Y}{40000}))$$

Place Bet

$$W_{i,m+1,j,k} = W_{i,m,j,k} + Y$$

$$W_{i,m,j,k+1} = W_{i,m,j,k} + \pi_{i,m,j,k}$$

$$\lambda_{k+1} = \begin{cases} \lambda_0 & \text{if } w_k < p \\ \min(1, c\lambda_k) & \text{if } w_k \ge p \end{cases} \qquad \pi_k = \begin{cases} B_k m & \text{if } w_k < p \\ -B_k & \text{if } w_k \ge p \end{cases}$$

$$w_k \sim \text{Uniform}(0,1) \quad k = 1, 2, \ldots, n_{i,m,j}$$

$$B_k = \min(B_{max}, B_0 \frac{\lambda_k}{\lambda_0})$$

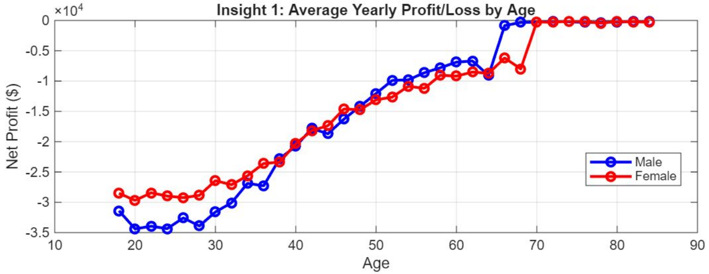
$$B_0 = 0.001Y \quad B_{max} = 0.05Y$$

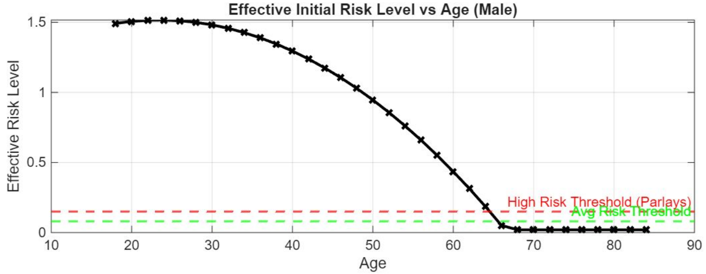
Calculate Total Profits

$$P = \frac{1}{N} \sum_{i=1}^{N} \sum_{m=1}^{12} \sum_{j=1}^{B_{feq}} \sum_{k=1}^{n_{i,m,j}} \pi_{i,m,j,k}$$

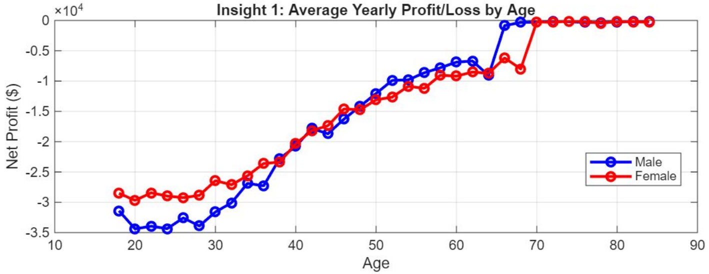
Calvin

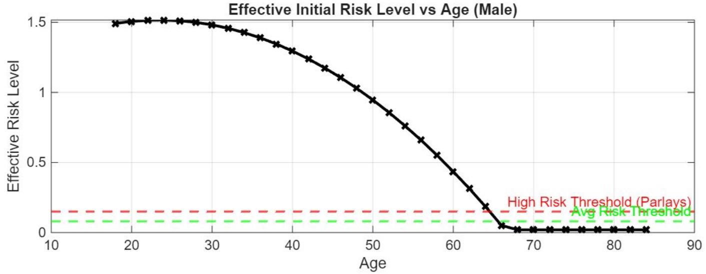
Psychological Factors in Financial Decision Making





Psychological Factors in Financial Decision Making

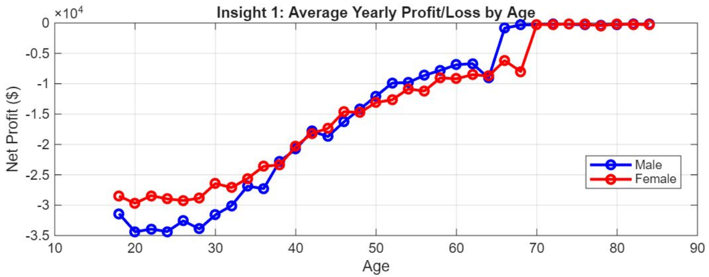


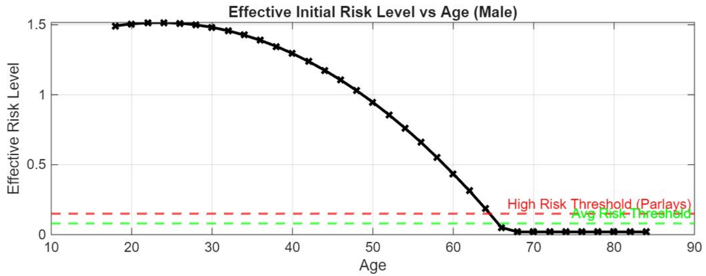


Youth Correlates with Financial Losses.

Calvin

Psychological Factors in Financial Decision Making

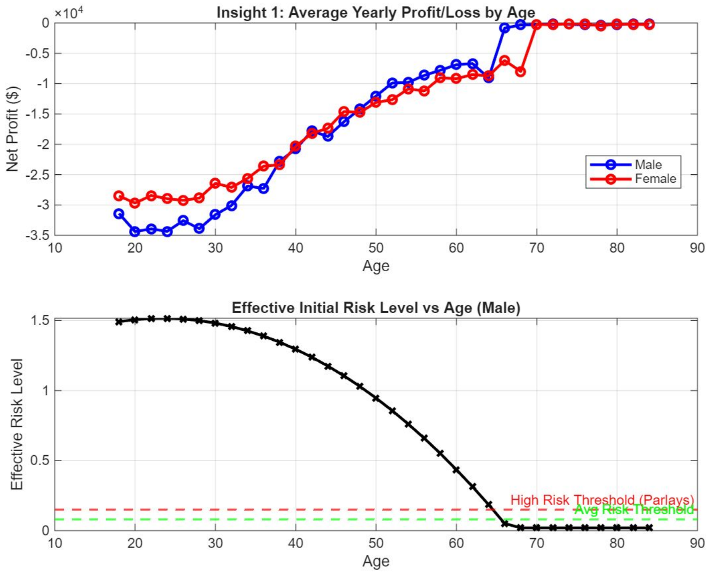


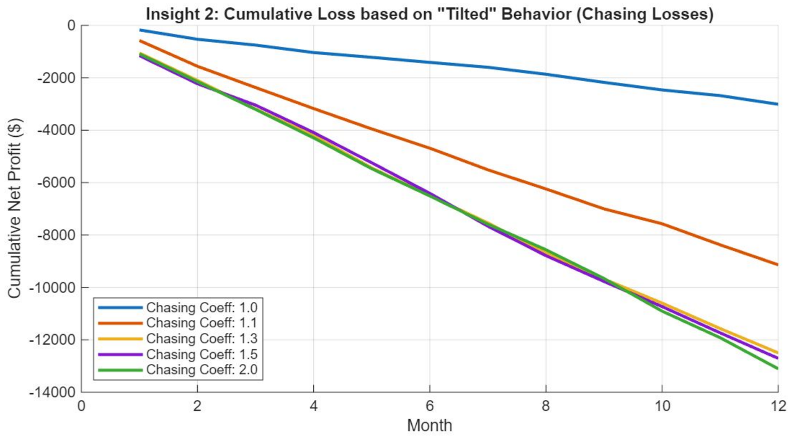


Youth Correlates with Financial Losses.

Extreme Risk Appetite in Younger Males: These steep early losses are driven by disproportionate risk-taking.

Psychological Factors in Financial Decision Making



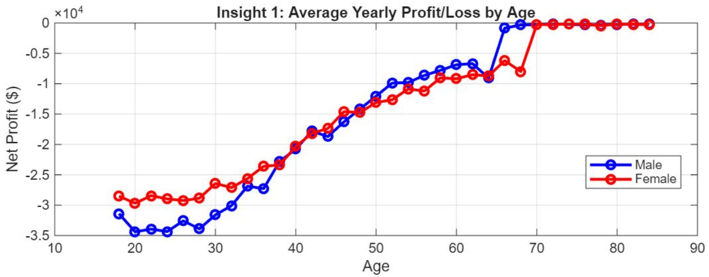


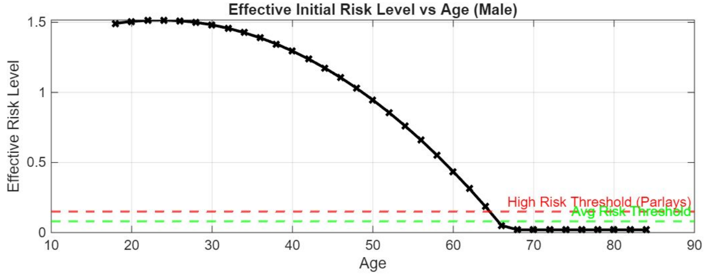
Youth Correlates with Financial Losses.

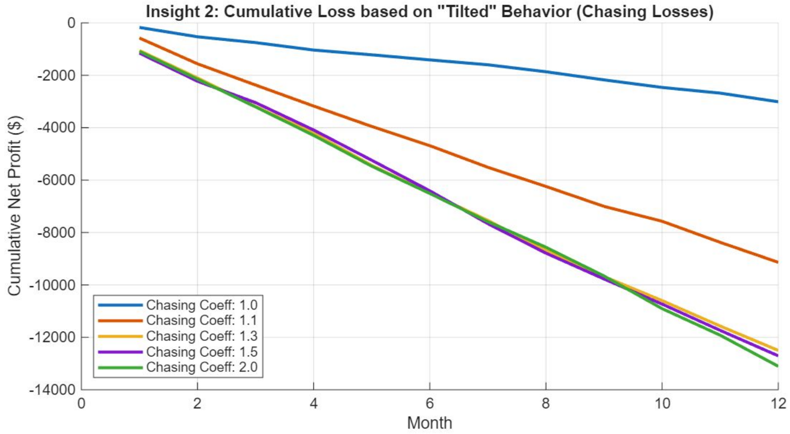
Extreme Risk Appetite in Younger Males: These steep early losses are driven by disproportionate risk-taking.

Calvin

Psychological Factors in Financial Decision Making







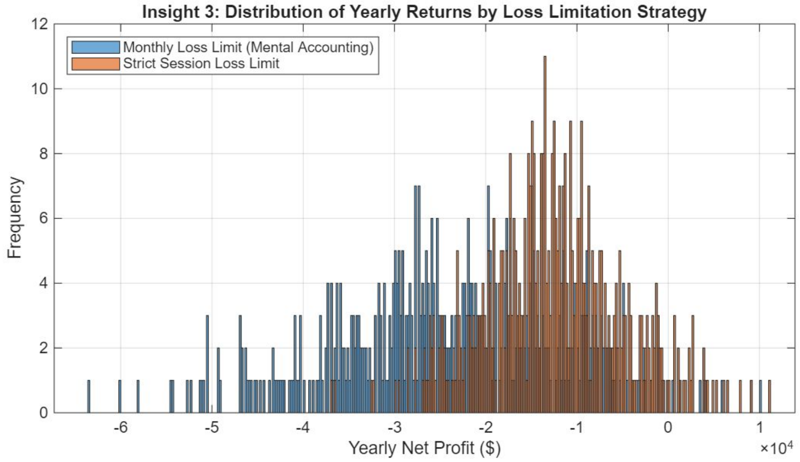
Impact of "Chasing Losses": Falling into "tilted" behavior drastically accelerates financial ruin.

Youth Correlates with Financial Losses.

Extreme Risk Appetite in Younger Males: These steep early losses are driven by disproportionate risk-taking.

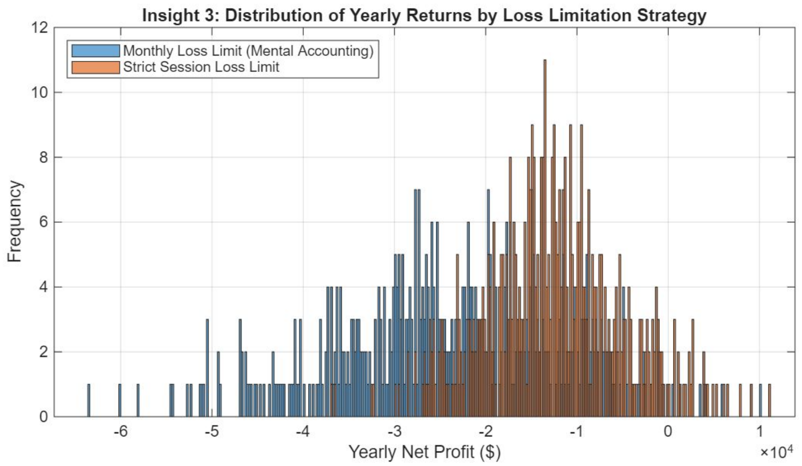
Financial Distribution & Income Impact Analysis

Stan



Financial Distribution & Income Impact Analysis

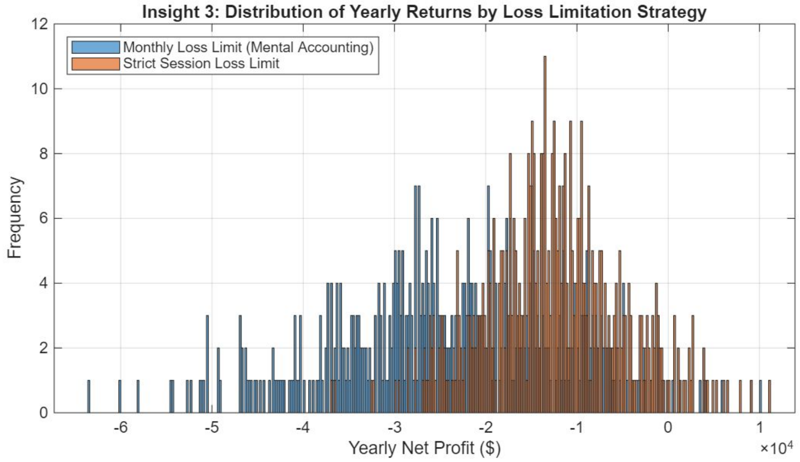
Stan



Strict Limits Prevent Extreme Ruin.

Financial Distribution & Income Impact Analysis

Stan

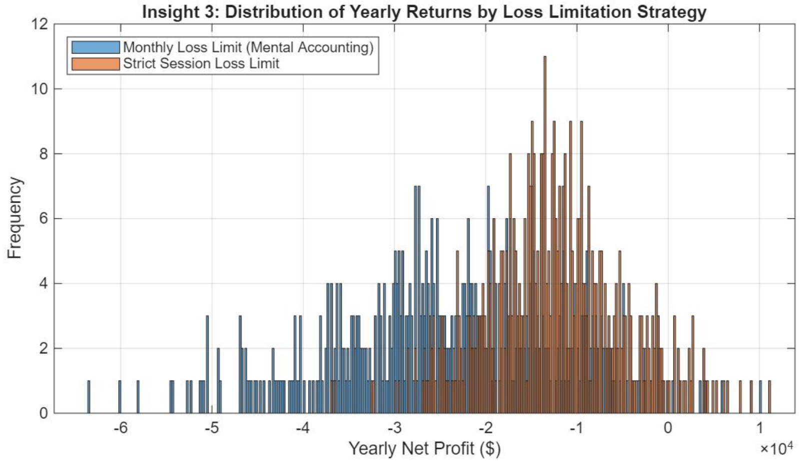


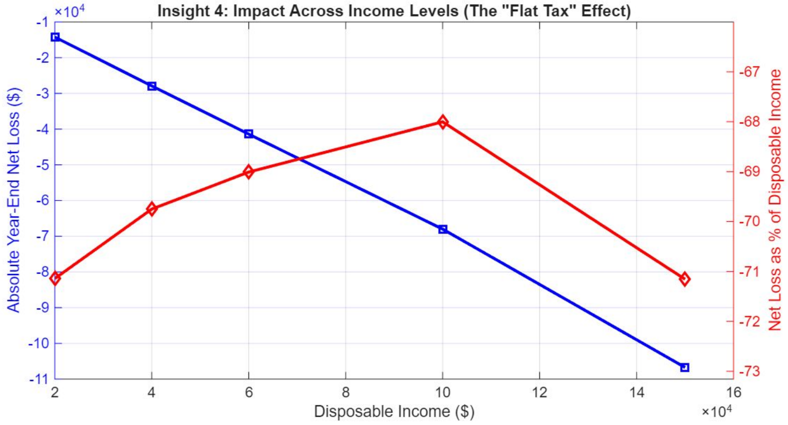
Strict Limits Prevent Extreme Ruin.

Relying on informal "monthly mental accounting" is highly ineffective.

Financial Distribution & Income Impact Analysis

Stan



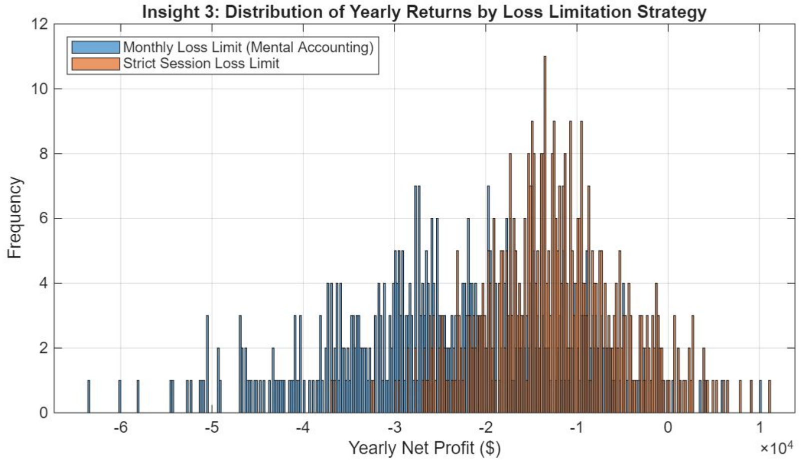


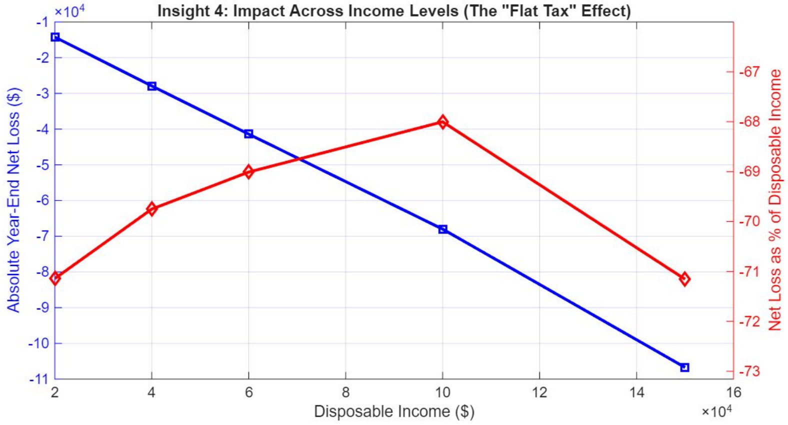
Strict Limits Prevent Extreme Ruin.

Relying on informal "monthly mental accounting" is highly ineffective.

Financial Distribution & Income Impact Analysis

Stan





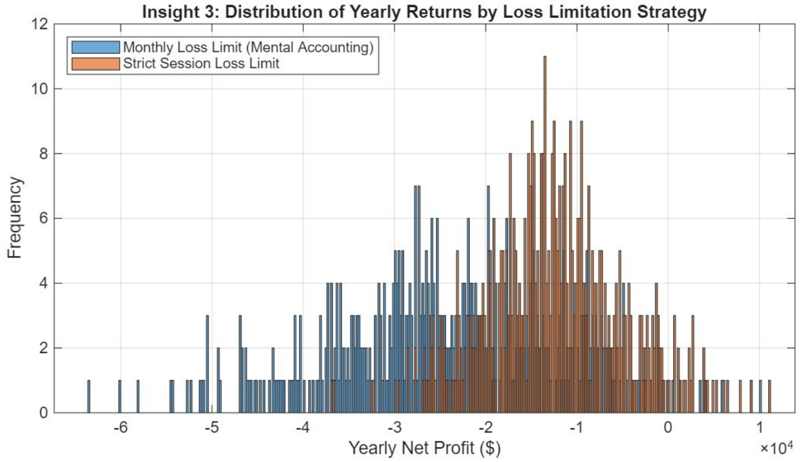
Strict Limits Prevent Extreme Ruin.

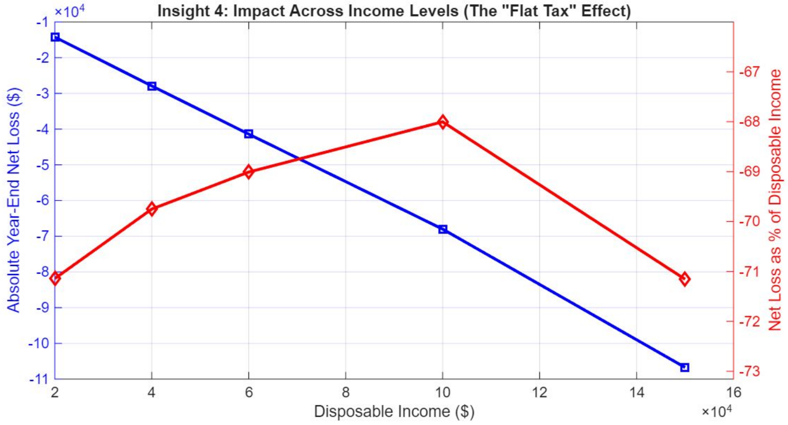
Relying on informal "monthly mental accounting" is highly ineffective.

Absolute Losses Scale with Wealth.

Financial Distribution & Income Impact Analysis

Stan





Strict Limits Prevent Extreme Ruin.

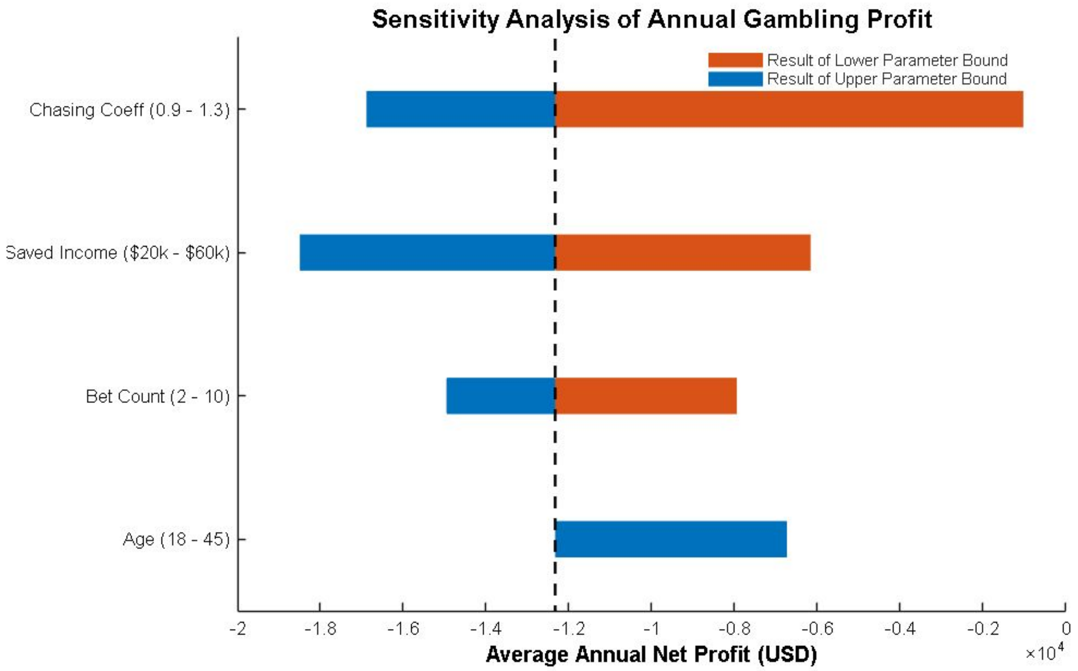
Relying on informal "monthly mental accounting" is highly ineffective.

Absolute Losses Scale with Wealth.

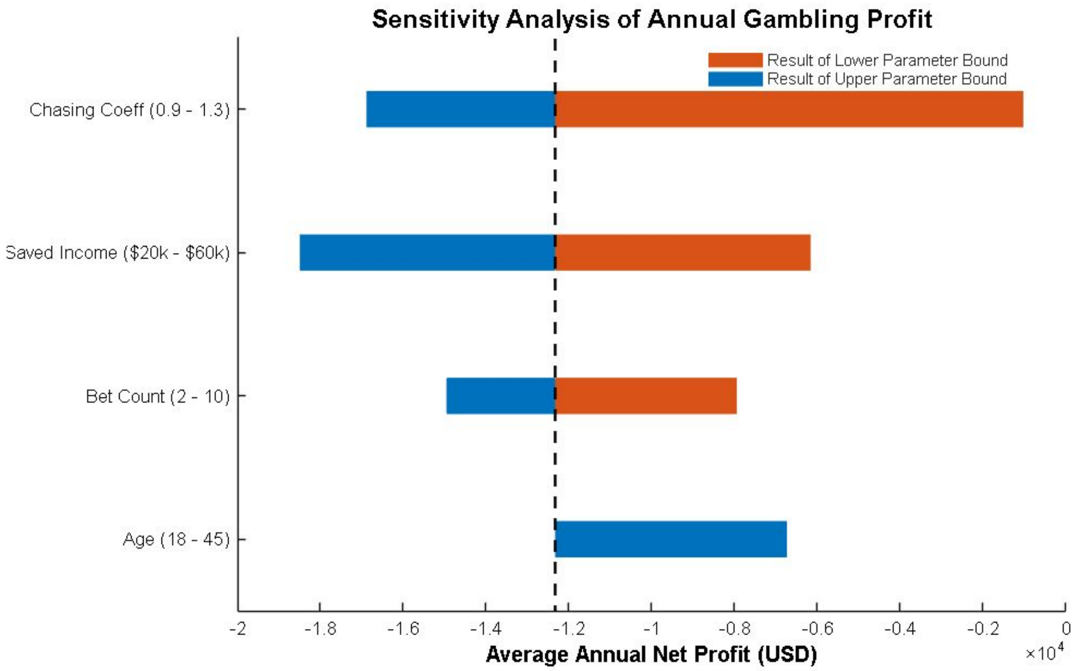
The "Flat Tax" Proportionality: Despite higher earners losing more total dollars, the relative financial damage is uniform.

Sensitivity And Scenario Analysis

Sensitivity And Scenario Analysis

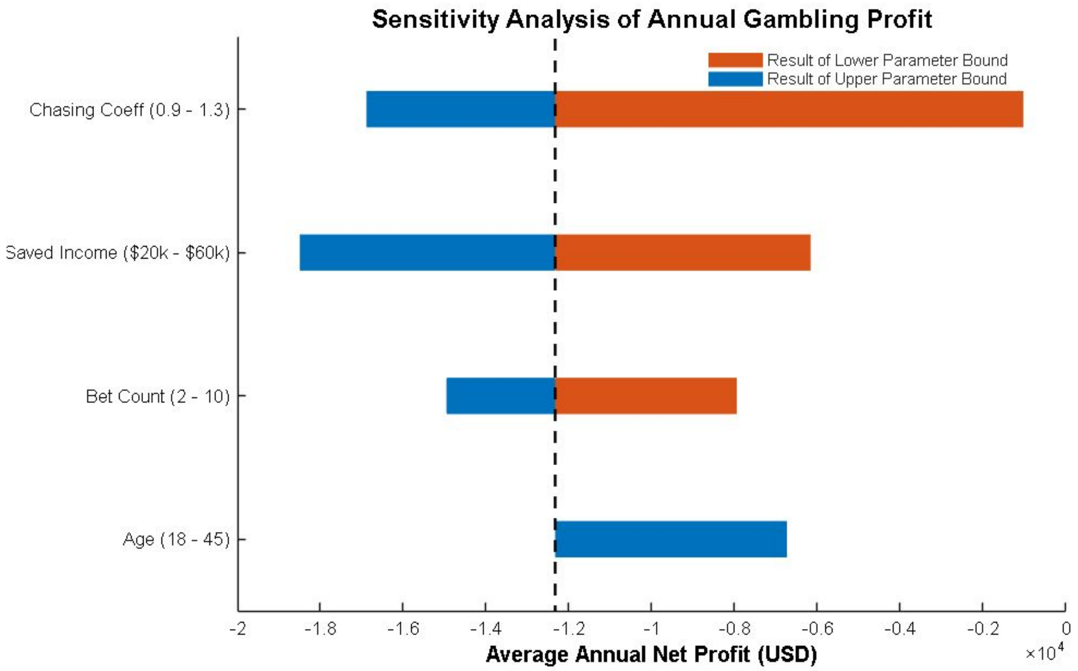


Sensitivity And Scenario Analysis



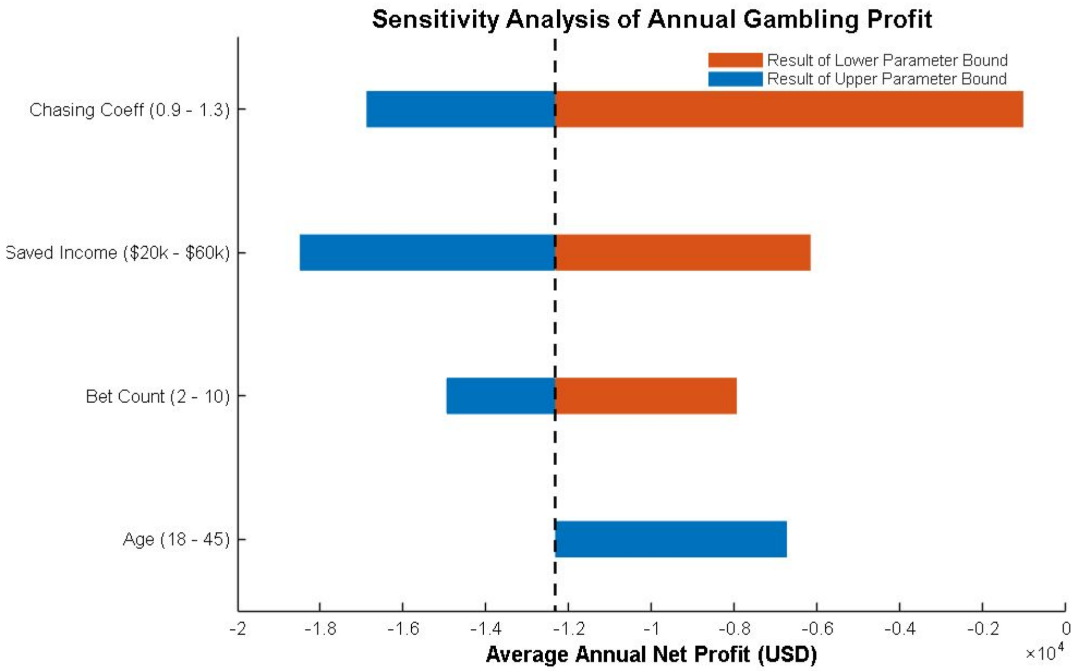
- Testing across typical gambler demographics confirms that our simulation...
  - scales smoothly

Sensitivity And Scenario Analysis



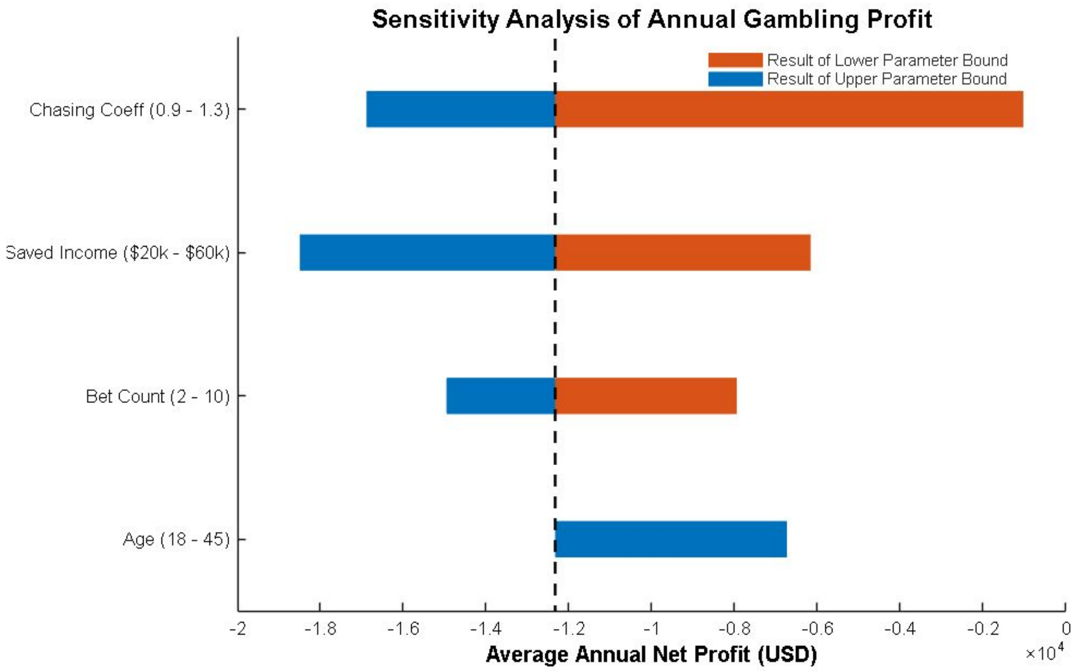
- Testing across typical gambler demographics confirms that our simulation...

Sensitivity And Scenario Analysis



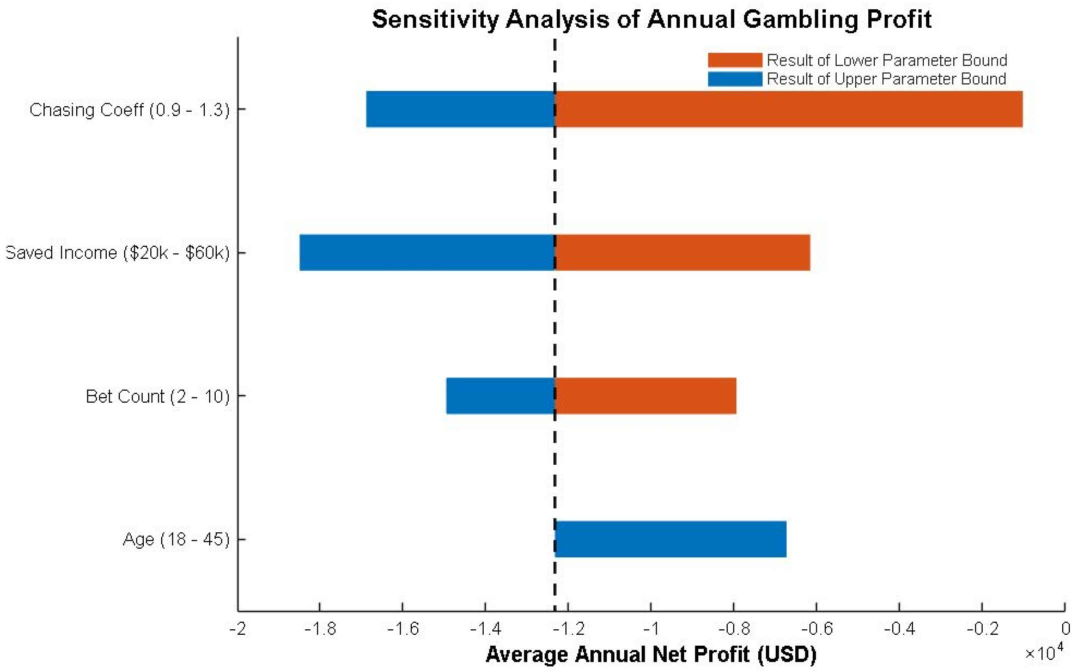
- Testing across typical gambler demographics confirms that our simulation...
  - scales smoothly
  - remains strictly bounded

Sensitivity And Scenario Analysis



- Testing across typical gambler demographics confirms that our simulation...
  - scales smoothly
  - remains strictly bounded
- We applied a Behavioral Stress Test to the Chasing Coefficient

Sensitivity And Scenario Analysis



- Testing across typical gambler demographics confirms that our simulation...
  - scales smoothly
  - remains strictly bounded
- We applied a Behavioral Stress Test to the Chasing Coefficient
  - flipping the psychology from 'loss-chasing' (1.3) to 'loss-averse' (0.9)

Don’t break the bank

Q3 - an introduction

Q3 - an introduction

- Model the long-run financial damage caused by gambling losses

Q3 - an introduction

- Model the long-run financial damage caused by gambling losses
    - Convert that damage into Recovery Burden Months (RBM)

Q3 - an introduction

- ❑ Model the long-run financial damage caused by gambling losses
  - ❑ Convert that damage into Recovery Burden Months (RBM)
  - ❑ Measure how many months of ordinary saving are needed to repair the harm

Q3 - an introduction

- Model the long-run financial damage caused by gambling losses
  - Convert that damage into Recovery Burden Months (RBM)
  - Measure how many months of ordinary saving are needed to repair the harm
- We built a 3-compartment recovery model:

Q3 - an introduction

- Model the long-run financial damage caused by gambling losses
  - Convert that damage into Recovery Burden Months (RBM)
  - Measure how many months of ordinary saving are needed to repair the harm
- We built a 3-compartment recovery model:
  - depleted savings

Q3 - an introduction

- Model the long-run financial damage caused by gambling losses
  - Convert that damage into Recovery Burden Months (RBM)
  - Measure how many months of ordinary saving are needed to repair the harm
- We built a 3-compartment recovery model:
  - depleted savings
  - missed investment growth

Q3 - an introduction

- Model the long-run financial damage caused by gambling losses
  - Convert that damage into Recovery Burden Months (RBM)
  - Measure how many months of ordinary saving are needed to repair the harm
- We built a 3-compartment recovery model:
  - depleted savings
  - missed investment growth
  - debt accumulation

What data did we use:

What data did we use:

- ❑ Typical low-risk fund returns (7%)

What data did we use:

- ❏ Typical low-risk fund returns (7%)
- ❏ Average credit card APR (22%)

Stan

What data did we use:

- ❏ Typical low-risk fund returns (7%)
- ❏ Average credit card APR (22%)
- ❏ Research journals on average gambling loss ($200pm)

Stan

What data did we use:

- ❑ Typical low-risk fund returns (7%)
- ❑ Average credit card APR (22%)
- ❑ Research journals on average gambling loss ($200pm)
    - ❑ From Normative Feedback intervention by gambling operator can decrease average wages per day.

Stan

What data did we use:

- Typical low-risk fund returns (7%)
- Average credit card APR (22%)
- Research journals on average gambling loss ($200pm)
    - From Normative Feedback intervention by gambling operator can decrease average wages per day.
- Impulse/confort spending ($50pm)

Stan

What data did we use:

- ❑ Typical low-risk fund returns (7%)
- ❑ Average credit card APR (22%)
- ❑ Research journals on average gambling loss ($200pm)
  - ❑ From Normative Feedback intervention by gambling operator can decrease average wages per day.
- ❑ Impulse/confort spending ($50pm)
  - ❑ Report by Self Financial on Americans' Mental Health and Personal Spending.
- ❑ Monthly saving ($500pm)

Stan

What data did we use:

- ❑ Typical low-risk fund returns (7%)
- ❑ Average credit card APR (22%)
- ❑ Research journals on average gambling loss ($200pm)
  - ❑ From Normative Feedback intervention by gambling operator can decrease average wages per day.
- ❑ Impulse/confort spending ($50pm)
  - ❑ Report by Self Financial on Americans' Mental Health and Personal Spending.
- ❑ Monthly saving ($500pm)
  - ❑ NatWest Saving Index

Model structure

Model structure

- ❏ Step 1: compute the harmful shortfall

Model structure

- Step 1: compute the harmful shortfall
  - $q = \max(0,\ g_{\text{monthly}} + s_{\text{monthly}} - c_{\text{monthly}})$

Model structure

- Step 1: compute the harmful shortfall
  - $q = \max(0,\ g_{\text{monthly}} + s_{\text{monthly}} - c_{\text{monthly}})$
- Step 2: convert annual rates to monthly rates

Model structure

- Step 1: compute the harmful shortfall
  - $q = \max(0,\ g_{\text{monthly}} + s_{\text{monthly}} - c_{\text{monthly}})$
- Step 2: convert annual rates to monthly rates
- Step 3: solve the 3-gap ODE system with MATLAB ode45

Model structure

- Step 1: compute the harmful shortfall
  - $$q = \max(0,\ g_{\text{monthly}} + s_{\text{monthly}} - c_{\text{monthly}})$$
- Step 2: convert annual rates to monthly rates
- Step 3: solve the 3-gap ODE system with MATLAB ode45
- Step 4: extract final gaps and convert them into RBM

Execution

Execution

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\mathrm{rep}}}$$

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\mathrm{rep}}}$$

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\mathrm{rep}}}$$

Monthly saving capacity

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\mathrm{rep}}}$$

Monthly saving capacity

$$q = \max(0,\ g_{\mathrm{monthly}} + s_{\mathrm{monthly}} - c_{\mathrm{monthly}})$$

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\mathrm{rep}}}$$

Monthly saving capacity

Harmful shortfall beyond safe cushion

$$q = \max(0,\ g_{\mathrm{monthly}} + s_{\mathrm{monthly}} - c_{\mathrm{monthly}})$$

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\mathrm{rep}}}$$

Monthly saving capacity

Harmful shortfall beyond safe cushion

$$q = \max(0,\ g_{\mathrm{monthly}} + s_{\mathrm{monthly}} - c_{\mathrm{monthly}})$$

$$\frac{d\Delta_A}{dt} = aq, \qquad \frac{d\Delta_W}{dt} = r\Delta_W + wq, \qquad \frac{d\Delta_B}{dt} = i_d\Delta_B + bq$$

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\mathrm{rep}}}$$

Monthly saving capacity

Harmful shortfall beyond safe cushion

$$q = \max(0,\ g_{\mathrm{monthly}} + s_{\mathrm{monthly}} - c_{\mathrm{monthly}})$$

Depleted savings

$$\frac{d\Delta_A}{dt} = aq, \qquad \frac{d\Delta_W}{dt} = r\Delta_W + wq, \qquad \frac{d\Delta_B}{dt} = i_d\Delta_B + bq$$

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\mathrm{rep}}}$$

Monthly saving capacity

Harmful shortfall beyond safe cushion

$$q = \max(0,\ g_{\mathrm{monthly}} + s_{\mathrm{monthly}} - c_{\mathrm{monthly}})$$

Depleted savings

Missed investment growth

$$\frac{d\Delta_A}{dt} = aq, \qquad \frac{d\Delta_W}{dt} = r\Delta_W + wq, \qquad \frac{d\Delta_B}{dt} = i_d\Delta_B + bq$$

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\mathrm{rep}}}$$

Monthly saving capacity

Harmful shortfall beyond safe cushion

$$q = \max(0,\ g_{\mathrm{monthly}} + s_{\mathrm{monthly}} - c_{\mathrm{monthly}})$$

Depleted savings

Missed investment growth

Debt accumulation

$$\frac{d\Delta_A}{dt} = aq, \qquad \frac{d\Delta_W}{dt} = r\Delta_W + wq, \qquad \frac{d\Delta_B}{dt} = i_d\Delta_B + bq$$

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\text{rep}}}$$

Monthly saving capacity

Harmful shortfall beyond safe cushion

$$q = \max(0,\ g_{\text{monthly}} + s_{\text{monthly}} - c_{\text{monthly}})$$

Depleted savings

Missed investment growth

Debt accumulation

$$\frac{d\Delta_A}{dt} = aq, \qquad \frac{d\Delta_W}{dt} = r\Delta_W + wq, \qquad \frac{d\Delta_B}{dt} = i_d\Delta_B + bq$$

$$r = \frac{r_{\text{annual}}}{12}, \qquad i_d = \frac{i_{d,\text{annual}}}{12}, \qquad \Delta_A(0) = \Delta_W(0) = \Delta_B(0) = 0$$

Execution

Total repair deficit

$$\mathrm{RBM} = \frac{\Delta_A(T)+\Delta_W(T)+\Delta_B(T)+M}{C_{\text{rep}}}$$

Monthly saving capacity

Harmful shortfall beyond safe cushion

$$q = \max(0,\ g_{\text{monthly}} + s_{\text{monthly}} - c_{\text{monthly}})$$

Depleted savings

Missed investment growth

Debt accumulation

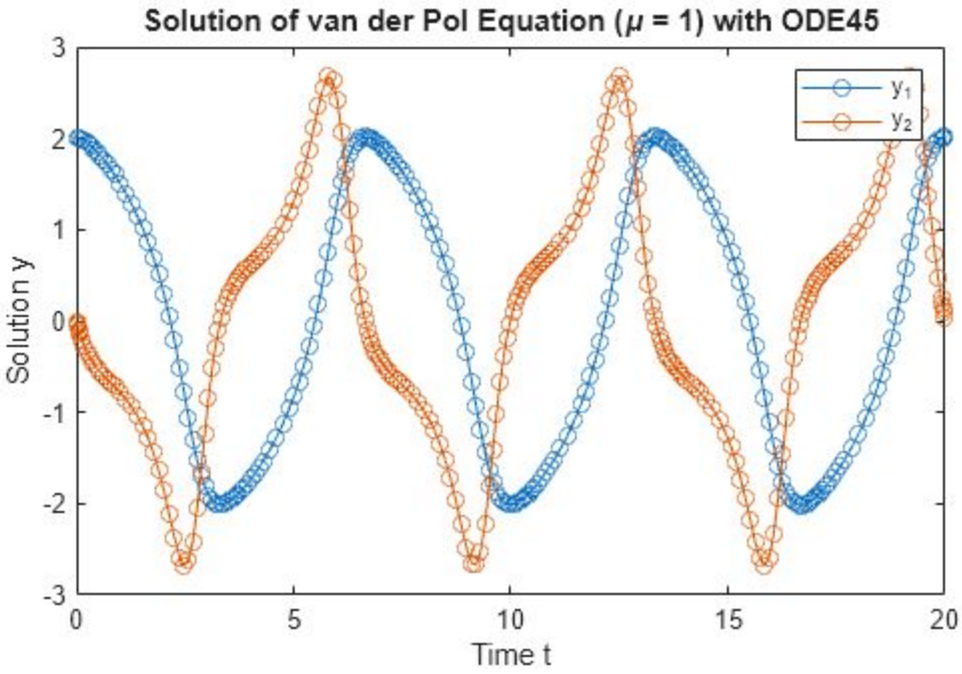
$$\frac{d\Delta_A}{dt} = aq, \qquad \frac{d\Delta_W}{dt} = r\Delta_W + wq, \qquad \frac{d\Delta_B}{dt} = i_d\Delta_B + bq$$

$$r = \frac{r_{\text{annual}}}{12}, \qquad i_d = \frac{i_{d,\text{annual}}}{12}, \qquad \Delta_A(0) = \Delta_W(0) = \Delta_B(0) = 0$$

Monthly rates + initial conditions

ode45 solver

ode45 solver



ode45 solver

```
% Monthly rates
r = r_annual / 12;
id = id_annual / 12;

% Harmless shortfall
q = max(0, g_monthly + s_monthly - c_monthly);

% ODE parameters
tspan = [0, months];
y0 = [0; 0; 0]; % Initial gaps: A, W, B

% Run ODE solver
[~, y] = ode45(@(t, y) gap_dynamics(t, y, q, a, w, b, r, id), tspan, y0);

% Final values
delta_A = y(end, 1);
delta_W = y(end, 2);
delta_B = y(end, 3);

% Total repair deficit
R = delta_A + delta_W + delta_B + M;

% Recovery Burden Months
RBM = R / C_rep;
```

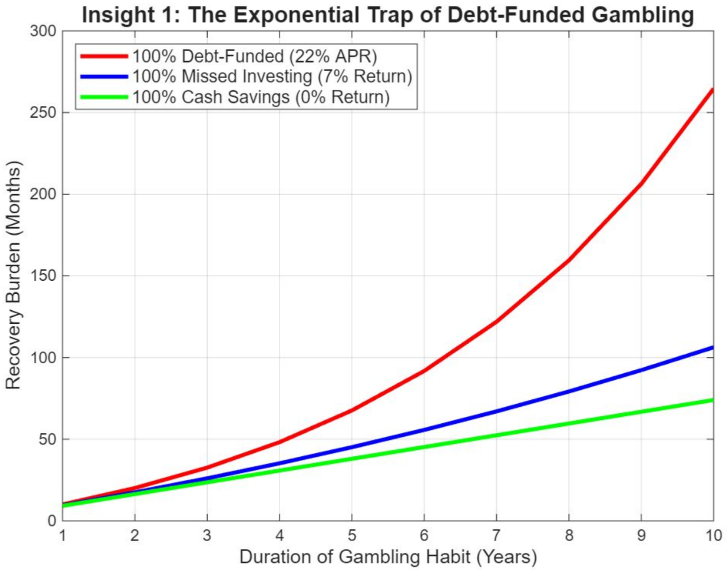
ode45 solver

```
% Monthly rates
r = r_annual / 12;
id = id_annual / 12;

% Harmless shortfall
q = max(0, g_monthly + s_monthly - c_monthly);

% ODE parameters
tspan = [0, months];
y0 = [0; 0; 0]; % Initial gaps: A, W, B

% Run ODE solver
[~, y] = ode45(@(t, y) gap_dynamics(t, y, q, a, w, b, r, id), tspan, y0);

% Final values
delta_A = y(end, 1);
delta_W = y(end, 2);
delta_B = y(end, 3);

% Total repair deficit
R = delta_A + delta_W + delta_B + M;

% Recovery Burden Months
RBM = R / C_rep;
```

- ode45 computes the time path of the three financial gaps and returns their final values for RBM

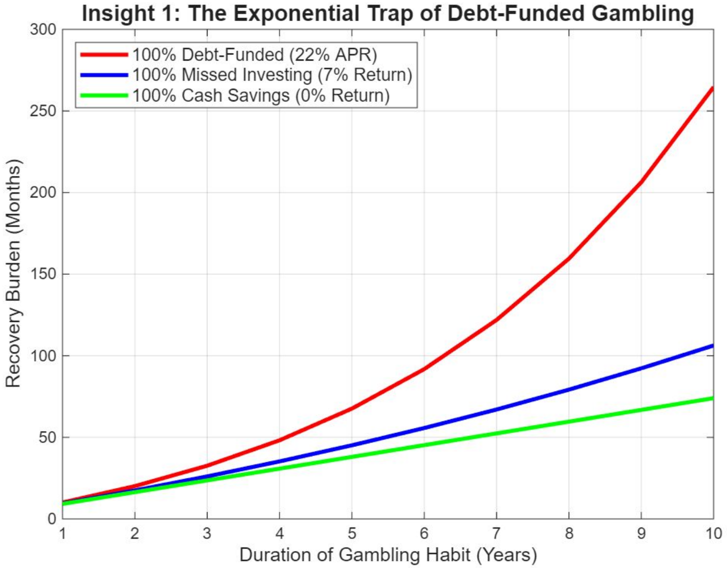
Gambling Impact & Recovery Analysis

Gambling Impact & Recovery Analysis



Gambling Impact & Recovery Analysis

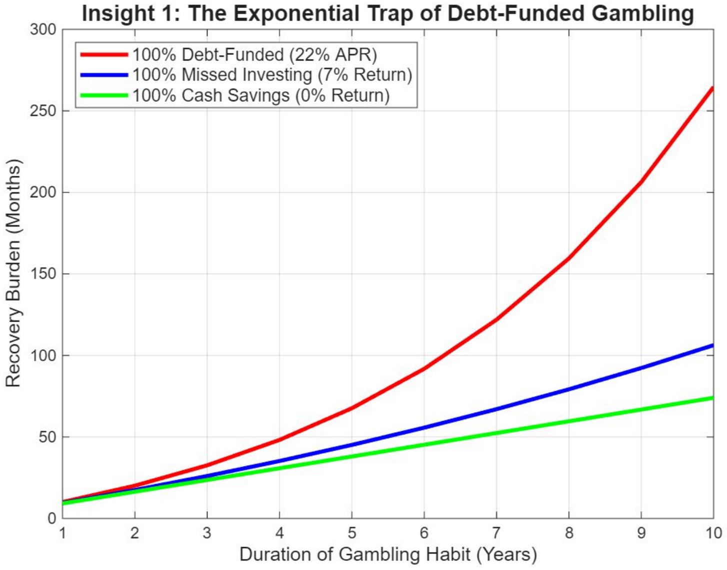
Calvin



The Exponential Trap of Debt: Funding gambling with high-interest debt creates an exponential recovery timeline.

Calvin

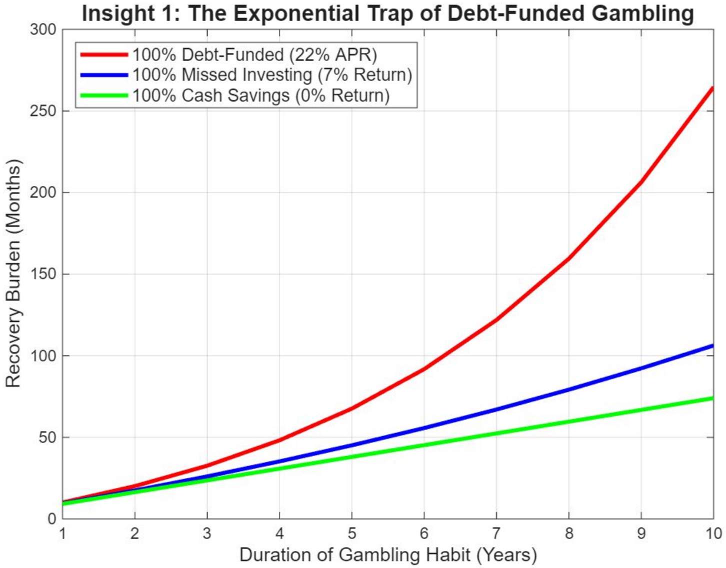
Gambling Impact & Recovery Analysis



The Exponential Trap of Debt: Funding gambling with high-interest debt creates an exponential recovery timeline.

Gambling Impact & Recovery Analysis

Calvin

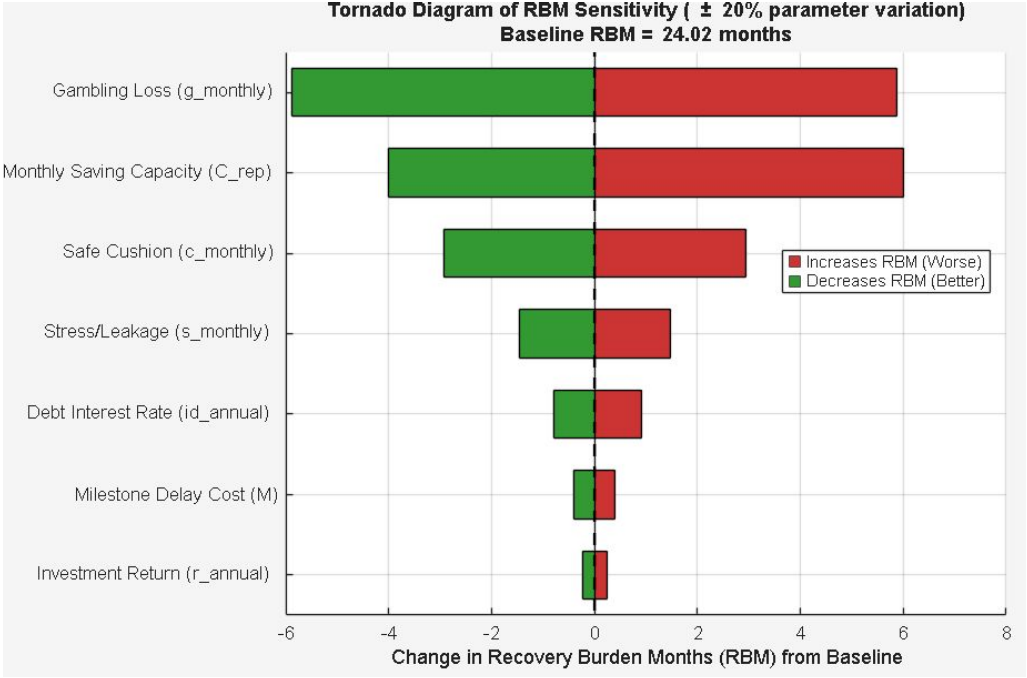


The Exponential Trap of Debt: Funding gambling with high-interest debt creates an exponential recovery timeline.

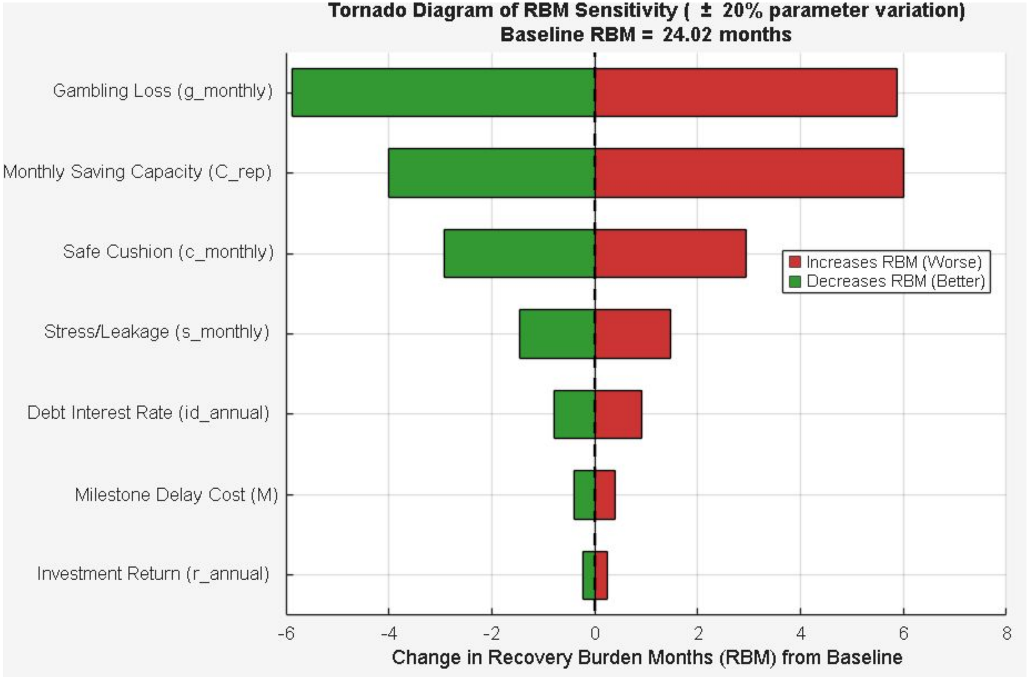
The "Safe Cushion" Illusion: Budgeting a financial buffer offers a false sense of security if limits are breached.

Sensitivity Analysis

Sensitivity Analysis

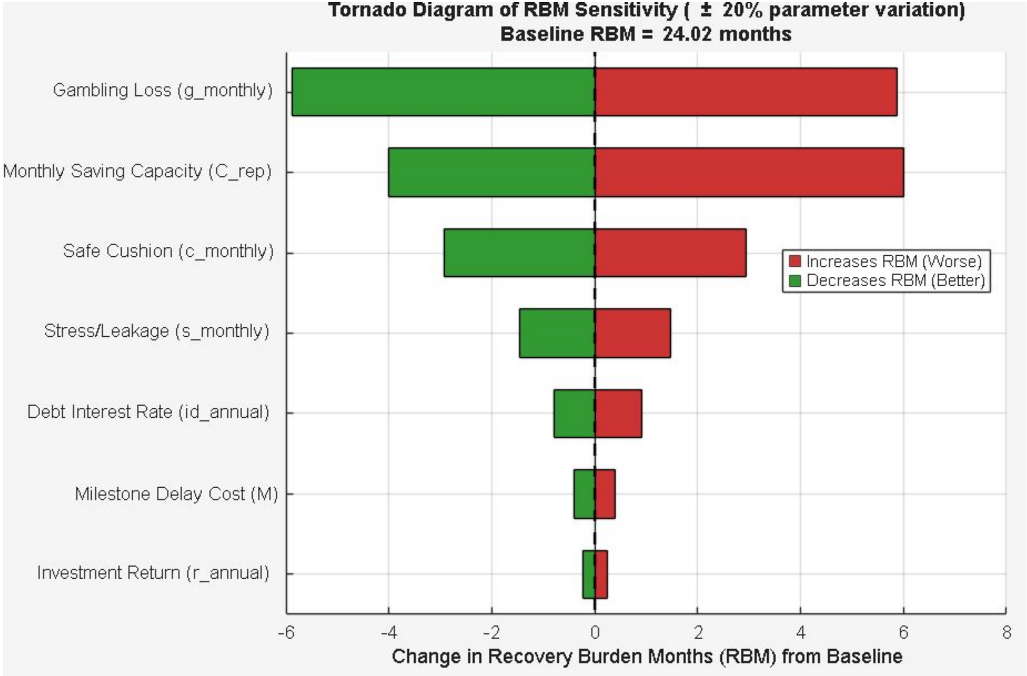


Sensitivity Analysis



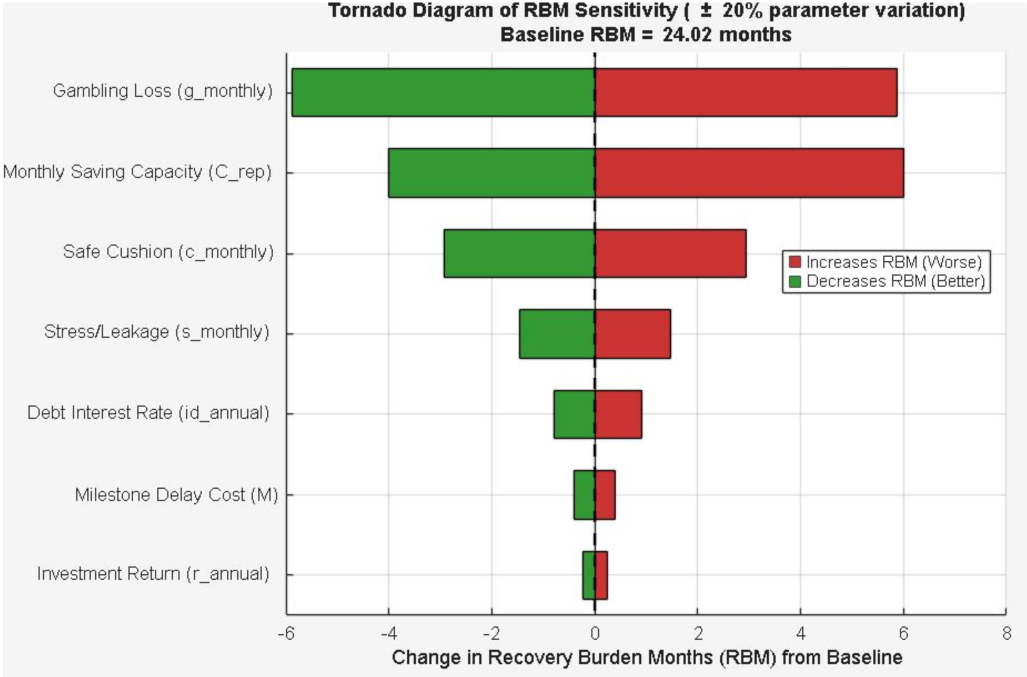
- **Insensitive to Rate Errors:**

Sensitivity Analysis



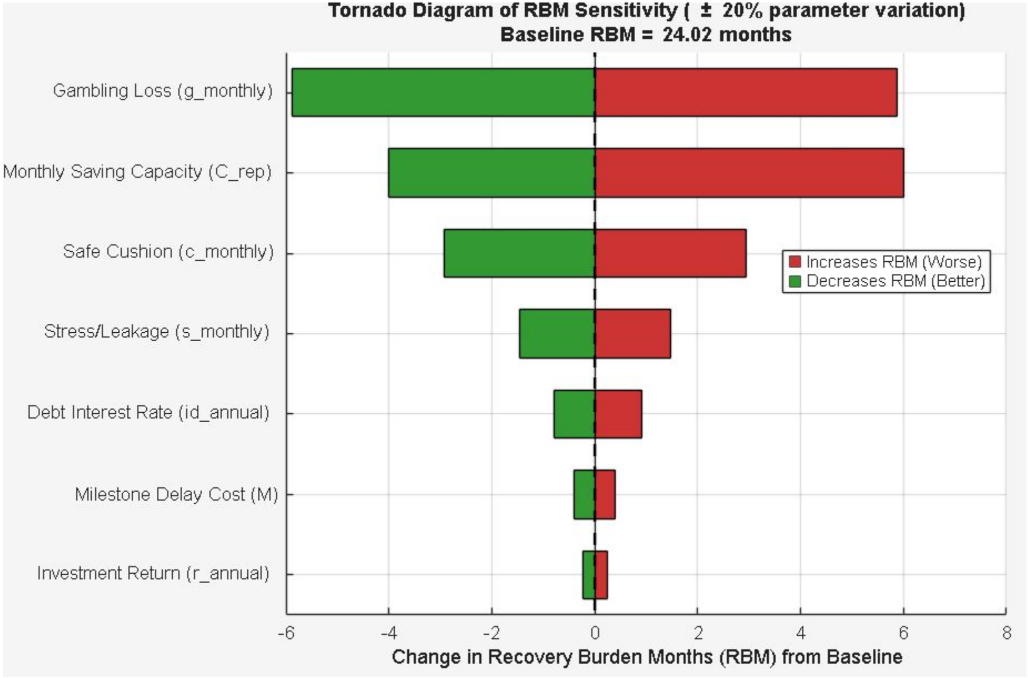
- **Insensitive to Rate Errors:**
  - Perfect APR or macroeconomic estimates not necessary for the RBM predictions

Sensitivity Analysis



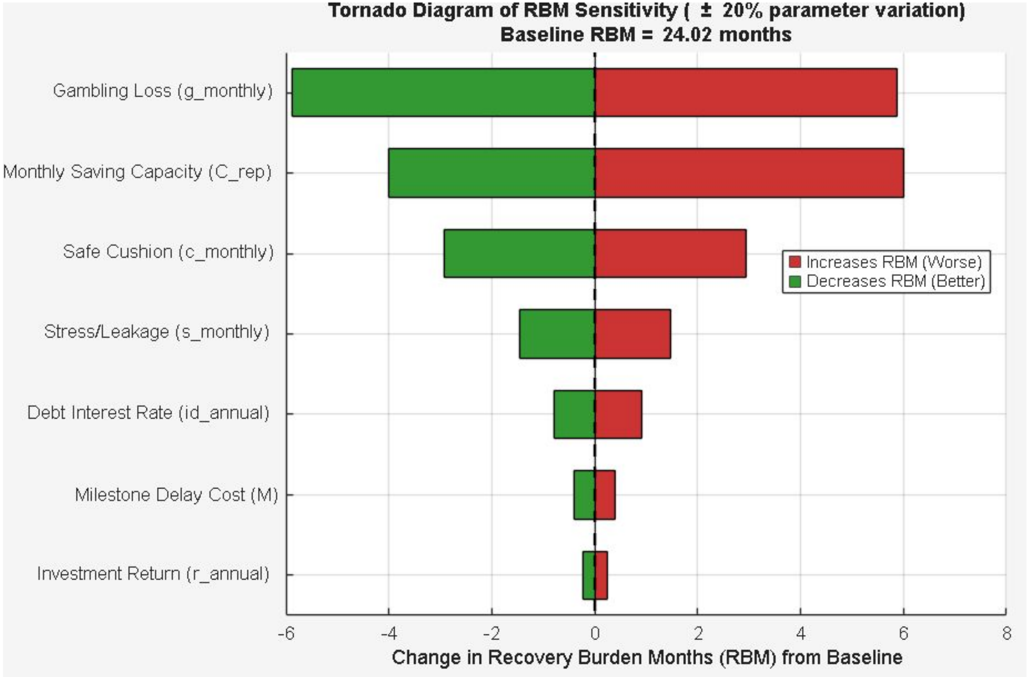
- **Insensitive to Rate Errors:**
  - Perfect APR or macroeconomic estimates not necessary for the RBM predictions
- **"Cash Flow is King":**

Sensitivity Analysis



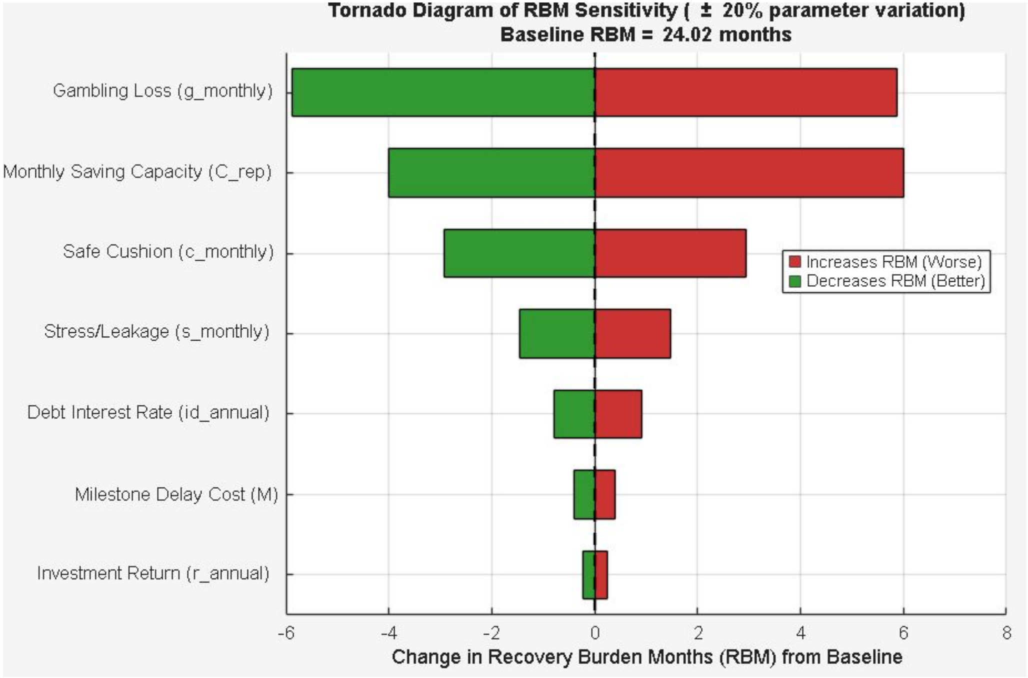
- **Insensitive to Rate Errors:**
  - Perfect APR or macroeconomic estimates not necessary for the RBM predictions
- **"Cash Flow is King":**
  - Recovery is dictated by four direct monthly cash flows:

Sensitivity Analysis



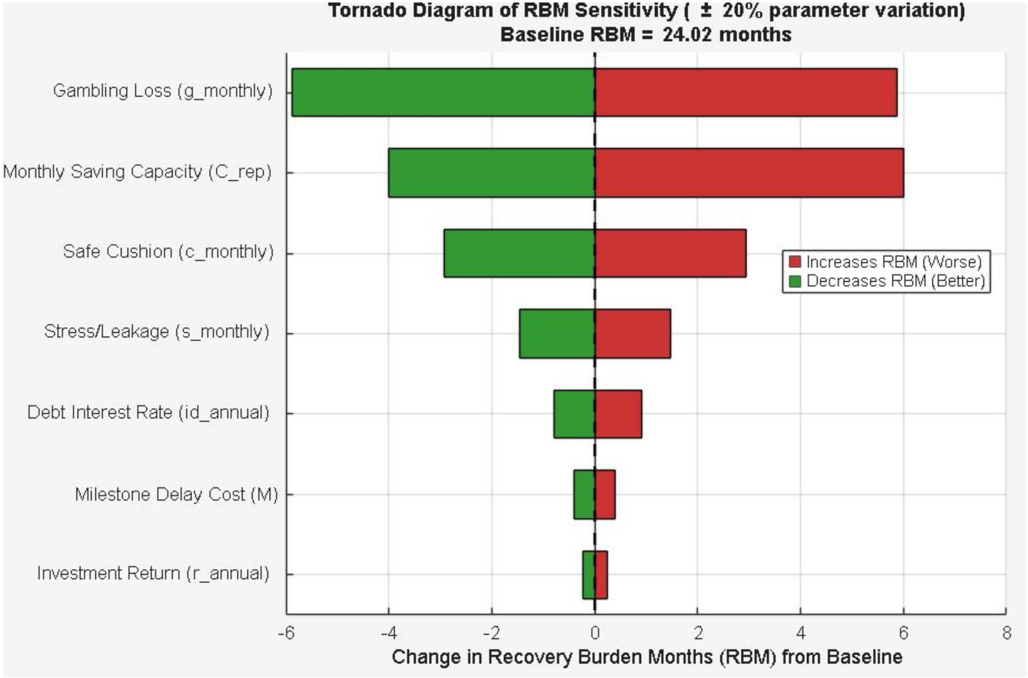
- **Insensitive to Rate Errors:**
  - Perfect APR or macroeconomic estimates not necessary for the RBM predictions
- **"Cash Flow is King":**
  - Recovery is dictated by four direct monthly cash flows:
    - Gambling Loss

Sensitivity Analysis



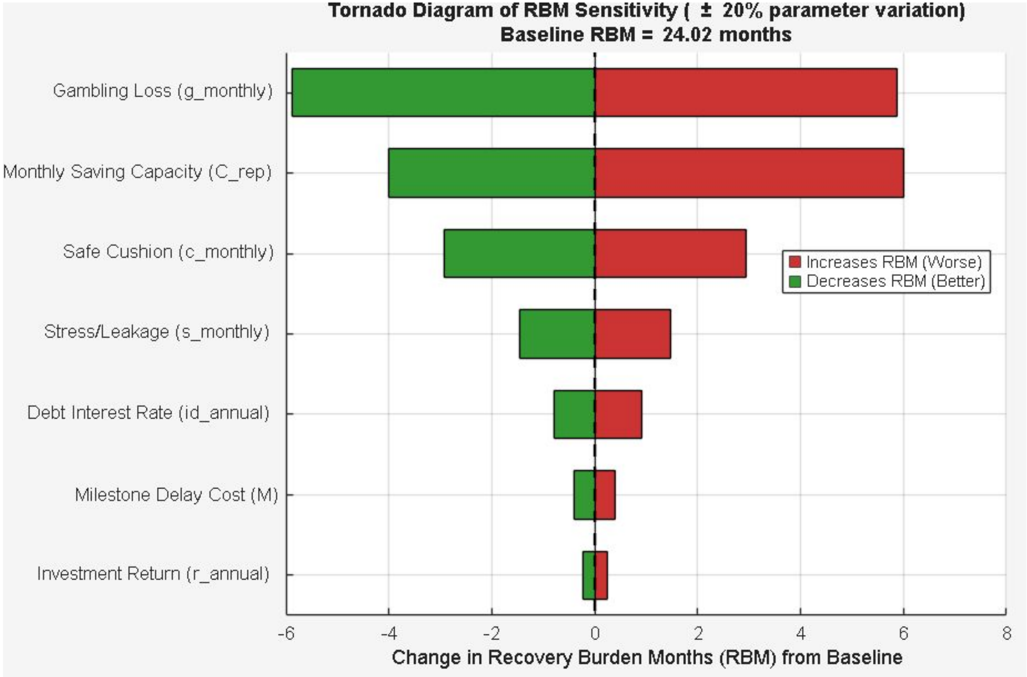
- **Insensitive to Rate Errors:**
  - Perfect APR or macroeconomic estimates not necessary for the RBM predictions
- **"Cash Flow is King":**
  - Recovery is dictated by four direct monthly cash flows:
    - Gambling Loss
    - Saving Capacity

Sensitivity Analysis



- **Insensitive to Rate Errors:**
  - Perfect APR or macroeconomic estimates not necessary for the RBM predictions
- **"Cash Flow is King":**
  - Recovery is dictated by four direct monthly cash flows:
    - Gambling Loss
    - Saving Capacity
    - Safe Cushion

Sensitivity Analysis



- **Insensitive to Rate Errors:**
  - Perfect APR or macroeconomic estimates not necessary for the RBM predictions
- **"Cash Flow is King":**
  - Recovery is dictated by four direct monthly cash flows:
    - Gambling Loss
    - Saving Capacity
    - Safe Cushion
    - Leakage

Findings

(backed by figures shown in the model)

Findings

(backed by figures shown in the model)

- ❑ Vulnerability exists before gambling begins

Calvin

Findings

(backed by figures shown in the model)

- ❑ Vulnerability exists before gambling begins
- ❑ Loss-chasing and risky betting behaviour amplify harm

Calvin

Findings

(backed by figures shown in the model)

- ❑ Vulnerability exists before gambling begins
- ❑ Loss-chasing and risky betting behaviour amplify harm
- ❑ Debt-funded losses create the worst long-run outcomes

Calvin

Findings

(backed by figures shown in the model)

- ❑ Vulnerability exists before gambling begins
- ❑ Loss-chasing and risky betting behaviour amplify harm
- ❑ Debt-funded losses create the worst long-run outcomes

<u>Overall</u>: Gambling harm is uneven, self-reinforcing and long-lasting

Technical computing!

Technical computing!

Stochastic simulation

Randomness

Technical computing!

Stochastic simulation	Randomness
Dynamic state updates	Behaviour changes

Technical computing!

Stochastic simulation	Randomness
Dynamic state updates	Behaviour changes
Numerical ODE solving	Long-run account

Technical computing!

Stochastic simulation	Randomness
Dynamic state updates	Behaviour changes
Numerical ODE solving	Long-run account
Sensitivity analysis	Identifying the key drivers

Thank you!

MathWorks Math Modeling Challenge

Society for Industrial and Applied Mathematics