Buoyancy Algebra - StepbyStep Calculation Walkthrough

This document shows exactly how the demo computes the numbers you see in the console and in the results files. It uses the same tokenization, word lists, and formulas as the Python script.

Formulas

• **Tokens:** split on words and punctuation `[A-Za-z']+|[!?]`
• **Polarity (simple):** add +1 for each POS token, 1.2 for each NEG token; normalize by `max(6, len(tokens))` -> clamp to [1, +1].
• **Harm:** `(-polarity + 1) / 2` -> higher if polarity is negative.
• **Provocation p:** `0.55*(CAPS/N) + 0.15*(bangs/N) + 0.45*(absolutes/N)`, with N = number of tokens, clamped to [0,1].
• **Gate4 metric Pi:** `harm / (1 + p)`

Worked Examples

Example 1 - Criminals sentence

Input: `All criminals deserve life in jail!!!`
Tokens: `['All', 'criminals', 'deserve', 'life', 'in', 'jail', '!', '!', '!']`

Before calculations
• Negative word hits: `1`
• Positive word hits: `0`
• Provocation components: CAPS=`0`, EXCL/QUES=`3`, ABSOLUTES=`1`, N=`9`
• Polarity (simple_sentiment): `-0.133` (1=very negative ... +1=very positive)
• Harm = `(-polarity + 1) / 2` = **0.567**
• Provocation p = `0.55*(CAPS/N) + 0.15*(bangs/N) + 0.45*(absolutes/N)` = **0.100**
• Pi_before = `harm / (1 + p)` = **0.52**

After one R->C->F cycle
Output: `While some people convicted of crimes deserve serious sentences according to available evidence, rehabilitation and prevention can reduce harm.`
Tokens: `['While', 'some', 'people', 'convicted', 'of', 'crimes', 'deserve', 'serious', 'sentences', 'according', 'to', 'available', 'evidence', 'rehabilitation', 'and', 'prevention', 'can', 'reduce', 'harm']`
• Negative hits: `0` | Positive hits: `5`
• Provocation components: CAPS=`0`, EXCL/QUES=`0`, ABSOLUTES=`0`, N=`19`
• Polarity_after: `0.263` -> Harm_after = **0.368**
• Prov_after: **0.000**
• Pi_after = **0.37**

Pi = Pi_before Pi_after = 0.52 0.37 = 0.15

Example 2 - Authority sentence

Input: `You should never question authority.`

Tokens: `['You', 'should', 'never', 'question', 'authority']`

Before calculations
• Negative word hits: `1`
• Positive word hits: `0`
• Provocation components: CAPS=`0`, EXCL/QUES=`0`, ABSOLUTES=`1`, N=`5`
• Polarity (simple_sentiment): `-0.200` (1=very negative ... +1=very positive)
• Harm = `(-polarity + 1) / 2` = **0.600**
• Provocation p = `0.55*(CAPS/N) + 0.15*(bangs/N) + 0.45*(absolutes/N)` = **0.090**
• Pi_before = `harm / (1 + p)` = **0.55**

After one R->C->F cycle
Output: `While You should some question authority according to available evidence, rehabilitation and prevention can reduce harm.`
Tokens: `['While', 'You', 'should', 'some', 'question', 'authority', 'according', 'to', 'available', 'evidence', 'rehabilitation', 'and', 'prevention', 'can', 'reduce', 'harm']`
• Negative hits: `0` | Positive hits: `5`
• Provocation components: CAPS=`0`, EXCL/QUES=`0`, ABSOLUTES=`0`, N=`16`
• Polarity_after: `0.312` -> Harm_after = **0.344**
• Prov_after: **0.000**
• Pi_after = **0.34**

Pi = Pi_before Pi_after = 0.55 0.34 = 0.21

Notes

• The operators R, Cx, F *change the text*, so the token counts, polarity, and provocation change too - that's why Pi drops.
• Because the same functions are used for the demo and for this walkthrough, the numbers match (within rounding).
• You can paste any sentence into the same functions to reproduce the exact results for that sentence.