Problema do Corte Máximo (Maximum Cut)

Corte Máximo

- **Instância:** Um grafo G com peso w_{ij} não negativo em cada aresta ij.
- **Objetivo:** Obter um corte de peso máximo em G, ou seja, uma partição (A, B) do conjunto de vértices tal que seja máxima a soma dos pesos das arestas entre A e B.

Algoritmo MaxCut-Erdős(G, w) - 1967

1. **para cada** vértice v de G, coloque v em A com prob. $1/2$
2. **seja** $B = V(G) - A$
3. **retorne** a partição (A, B) de $V(G)$

MaxCut-Erdős é 0.5-aproximativo probabilístico

- Seja X o peso do corte (A, B) retornado pelo algoritmo

$$\mathbb{E}(X) = \sum_{ij \in E(G)} w_{ij} \cdot \mathbb{P}(ij \in (A, B)) = \sum_{ij \in E(G)} \frac{1}{2} \cdot w_{ij} \geq 0.5 \cdot opt(G, w)$$

MaxCut-Erdős-Desaleatorizado é 0.5-aproximativo

Algoritmo MaxCut-Erdős-Desaleatorizado(G, w)

1. $A \leftarrow \emptyset$; $\quad B \leftarrow \emptyset$
2. **para** cada vértice $v \in V(G)$
3. $\quad$ **se** $EspCond(A \cup \{v\}, B) \geq EspCond(A, B \cup \{v\})$ **então**
4. $\quad\quad A \leftarrow A \cup \{v\}$
5. $\quad$ **senão** $B \leftarrow B \cup \{v\}$

Algoritmo $EspCond(G, w, A, B)$

1. $esp \leftarrow 0$
2. **para** cada aresta $ij \in E(G)$
3. $\quad$ **se** $(i \in A$ e $j \in B)$ ou $(i \in B$ e $j \in A)$ **então**
4. $\quad\quad esp \leftarrow esp + w_{ij}$
5. $\quad$ **senão se** $(i \notin A$ e $i \notin B)$ ou $(j \notin A$ e $j \notin B)$ **então**
6. $\quad\quad esp \leftarrow esp + w_{ij}/2$
7. **retorne** esp

MaxCut-Erdős-Desaleatorizado é 0.5-aproximativo

Algoritmo MaxCut-Erdős-Desaleatorizado(G, w)

1. $A \leftarrow \emptyset$; $\quad B \leftarrow \emptyset$
2. **para** cada vértice $v \in V(G)$
3. $\quad$ **se** $EspCond(A \cup \{v\}, B) \geq EspCond(A, B \cup \{v\})$ **então**
4. $\quad\quad A \leftarrow A \cup \{v\}$
5. $\quad$ **senão** $B \leftarrow B \cup \{v\}$

Observações

- MaxCut-Erdős-Desaleatorizado é 0.5-aproximativo (determinístico)
- Até 1995, esse era o melhor algoritmo aprox. p/ Problema MaxCut

MaxCut: Programação Quadrática Inteira

Formulação de Programação Quadrática Inteira

- Para cada $i \in V(G)$, usa-se um inteiro x_i
- Interpretação: $x_i = 1$ se $i \in A$; $x_i = -1$ se $i \in B$

- **Maximizar** $\frac{1}{2} \sum_{ij \in E(G)} w_{ij}(1 - x_i x_j)$, **restrito a**:
- Para cada $i \in V(G)$: $x_i \in \{-1, 1\}$

- É claramente viável; basta tomar $x_i = 1$ para todo $i \in V(G)$
- Obtém soℓ. exata $opt(G, w)$, mas problema NP-Difícil.

MaxCut: Progr. Quadrática Inteira (Relaxação vetorial)

Formulação de Programação Quadrática Inteira

- Para cada $i \in V(G)$, usa-se um inteiro x_i
- **Maximizar** $\frac{1}{2}\sum_{ij \in E(G)} w_{ij}(1 - x_i x_j)$, **restrito a**:
- Para cada $i \in V(G)$: $x_i x_i = 1$ (ou seja, $x_i \in \{-1, 1\}$)
- Obtém soℓ. exata $opt(G, w)$, mas problema NP-Difícil.

Relaxação vetorial da Programação Quadrática Inteira

- Para cada $i \in V(G)$, usa-se um vetor v_i de tam. n e de módulo 1
- $x_i = \pm 1 \quad \Rightarrow \quad v_i = [\pm 1, 0, \ldots, 0]$
- **Maximizar** $\frac{1}{2}\sum_{ij \in E(G)} w_{ij}(1 - v_i \cdot v_j)$, **restrito a**:
- Para cada $i \in V(G)$: $v_i \cdot v_i = 1$
- É viável; basta tomar $v_i = [1, 0, \ldots, 0]$ para todo $i \in V(G)$
- Obtém soℓ. relaxada $\geq opt(G, w)$.

MaxCut: Progr. Quadrática Inteira (Relaxação vetorial)

Observações

- Produto escalar (ou produto interno)
  $v_i \cdot v_j = \sum_{k=1}^{n} v_{ik} v_{jk} = v_{i1} v_{j1} + \ldots + v_{in} v_{jn}$
- Módulo (norma euclidiana) $|v_i| := \|v_i\|_2 = \sqrt{v_{i1}^2 + \ldots + v_{in}^2} = \sqrt{v_i v_i}$

Relaxação vetorial da Programação Quadrática Inteira

- Para cada $i \in V(G)$, usa-se um vetor v_i de tam. n e de módulo 1
- $x_i = \pm 1 \quad \Rightarrow \quad v_i = [\pm 1, 0, \ldots, 0]$

- **Maximizar** $\frac{1}{2} \sum_{ij \in E(G)} w_{ij}(1 - v_i \cdot v_j)$, **restrito a**:
- Para cada $i \in V(G)$: $\quad v_i \cdot v_i = 1$

- É viável; basta tomar $v_i = [1, 0, \ldots, 0]$ para todo $i \in V(G)$
- Obtém soℓ. relaxada $\geq opt(G, w)$.

MaxCut: Progr. Quadrática Inteira (Relaxação vetorial)

Relaxação vetorial da Programação Quadrática Inteira

- **Maximizar** $\frac{1}{2}\sum_{ij\in E(G)} w_{ij}(1 - v_i \cdot v_j)$, **restrito a**:
- Para cada $i \in V(G)$: $v_i \cdot v_i = 1$

Simplificação: encontrar vetores $v_1, \ldots, v_n$

- **Minimizar** $\sum_{ij\in E(G)} w_{ij}(v_i \cdot v_j)$, **restrito a**:
- Para cada $i \in V(G)$: $v_i \cdot v_i = 1$

Reformulação: achar matriz Y (linhas de Y são vetores v_i)

- **Minimizar** $\sum_{ij\in E(G)} w_{ij}(YY^T)_{ij}$, **restrito a**:
- Para cada $i \in V(G)$: $(YY^T)_{ii} = 1$

Reform: achar matriz X positiva semidefinida ($X = YY^T$)

- **Minimizar** $\sum_{ij\in E(G)} w_{ij}X_{ij}$, **restrito a**:
- Para cada $i \in V(G)$: $X_{ii} = 1$
- Existem algoritmos que obtém Y a partir de X positiva semidefinida e que resolvem programas gerais com matriz positiva semidefinida

Algoritmo MaxCut-GW (Goemans e Williamson, 1995)

Relaxação vetorial da Programação Quadrática Inteira

- **Maximizar** $\frac{1}{2}\sum_{ij\in E(G)} w_{ij}(1 - v_i \cdot v_j)$, **restrito a**:
- Para cada $i \in V(G)$: $\quad v_i \cdot v_i = 1$

Algoritmo MaxCut-GW(G, w)

1. **sejam** $v_1, \ldots, v_n$ uma solução ótima da Relaxação vetorial acima
2. **seja** $s \leftarrow$ *RandEsfera*(n) (vetor aleat. tam. n e módulo 1)
4. $A \leftarrow \{i \in V(G) : s \cdot v_i > 0\}$; $\qquad B \leftarrow V(G) - A$
5. **retorne** (A, B)

Lema: $\mathbb{P}(ij \in G_{AB}) = arccos(v_i \cdot v_j)/\pi$

Prova: $\mathbb{P}(ij \in G_{AB}) = \mathbb{P}(s \cdot v_i > 0, s \cdot v_j \leq 0) + \mathbb{P}(s \cdot v_i \leq 0, s \cdot v_j > 0)$

- $\mathbb{P}(s \cdot v_i > 0, s \cdot v_j \leq 0) = \frac{1}{2} \cdot \frac{\Theta_{ij}}{\pi}$, onde Θ_{ij} é o ângulo entre v_i e v_j (deve $\in$ região esfera interseção dos 2 semiespaços)
- $\Theta_{ij} = arccos(v_i \cdot v_j)$, $\quad$ pois $\quad v_i \cdot v_j = |v_i| \cdot |v_j| \cdot cos(\Theta_{ij})$

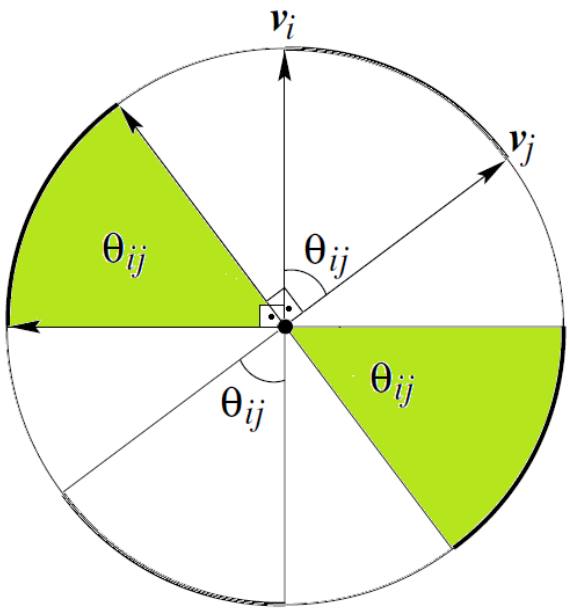
Algoritmo MaxCut-GW (Goemans e Williamson, 1995)

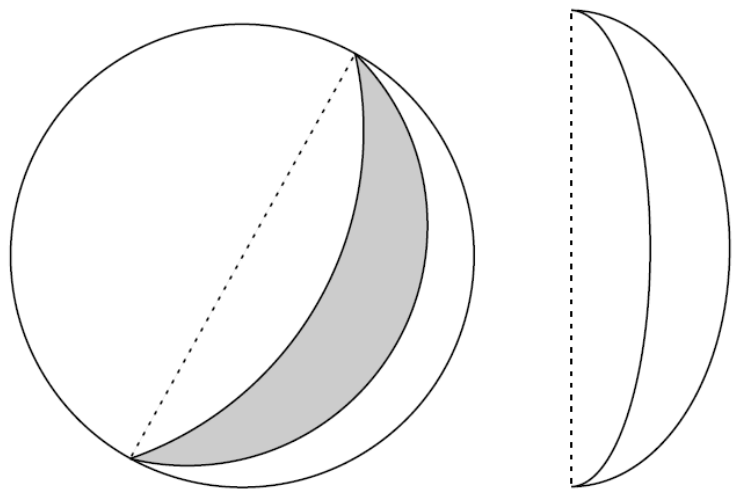
Relaxação vetorial da Programação Quadrática Inteira

- **Maximizar** $\frac{1}{2}\sum_{ij\in E(G)} w_{ij}(1 - v_i \cdot v_j)$, **restrito a**:
- Para cada $i \in V(G)$: $v_i \cdot v_i = 1$

Algoritmo MaxCut-GW(G, w)

1. **sejam** $v_1, \ldots, v_n$ uma solução ótima da Relaxação vetorial acima
2. **seja** $s \leftarrow$ *RandEsfera*(n) (vetor aleat. tam. n e módulo 1)
4. $A \leftarrow \{i \in V(G) : s \cdot v_i > 0\}$; $B \leftarrow V(G) - A$
5. **retorne** (A, B)





Uma "fatia" de uma esfera no $\mathbb{R}^3$

Algoritmo MaxCut-GW (Goemans e Williamson, 1995)

- **Maximizar** $\frac{1}{2}\sum_{ij\in E(G)} w_{ij}(1 - v_i \cdot v_j)$, **restrito a**:
- Para cada $i \in V(G)$: $v_i \cdot v_i = 1$

Algoritmo MaxCut-GW(G, w)

1. **sejam** $v_1, \ldots, v_n$ uma solução ótima da Relaxação vetorial acima
2. **seja** $s \leftarrow$ *RandEsfera*(n) (vetor aleat. tam. n e módulo 1)
4. $A \leftarrow \{i \in V(G) : s \cdot v_i > 0\}$; $B \leftarrow V(G) - A$
5. **retorne** (A, B)

Teorema: MaxCut-GW é 0.87856-aprox. probabilístico

Prova: Seja X o valor retornado por MaxCut-GW. Seja $E = E(G)$.

- $\mathbb{E}(X) = \sum_{ij\in E} w_{ij} \cdot \mathbb{P}(ij \in G_{AB}) = \sum_{ij\in E} w_{ij} \cdot \frac{1}{\pi} arccos(v_i \cdot v_j)$
- $\mathbb{E}(X) \geq \sum_{ij\in E} w_{ij} \cdot \frac{\alpha}{2}(1 - v_i \cdot v_j) \geq \alpha \cdot opt(G, w)$, onde

$$\alpha = \min_{x\in[-1,1]} \left\{ \frac{2 \cdot arccos(x)}{\pi(1-x)} \right\} = \min_{\Theta\in[0,\pi]} \left\{ \frac{2 \cdot \Theta}{\pi(1 - cos(\Theta))} \right\} = 0.87856$$

Algoritmo MaxCut-GW (Goemans e Williamson, 1995)

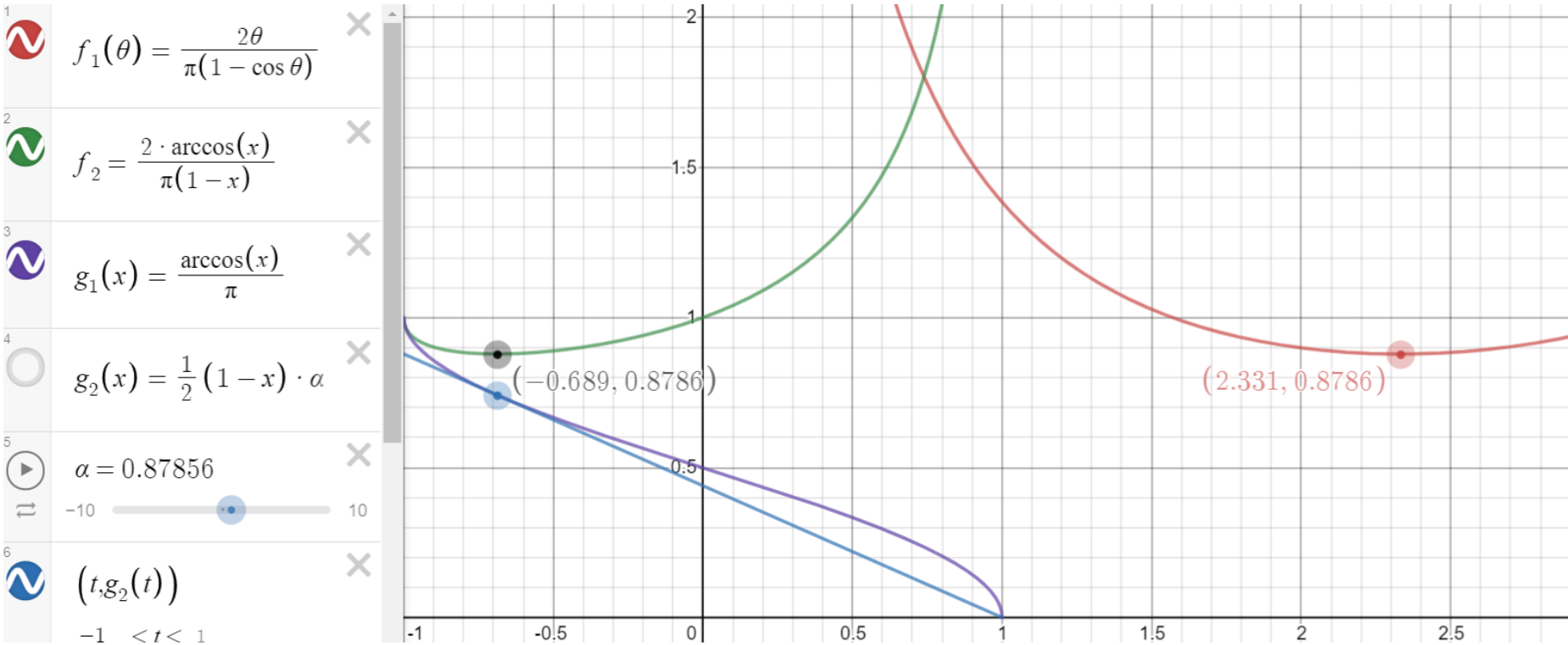


Figure: Gráficos no Desmos: $\alpha = 0.87856$

Conclusão: MaxCut

MaxCut

- 0.5-aprox: Algoritmo MaxCut-Erdős-Desaleatorizado (Erdős, 1967).
- 0.878-aprox prob.: MaxCut-GW (Goemans, Williamson, 1995)
- 0.878-aprox.: MaxCut-GW-Desaleat (Mahajan, Kamesh, 1995)
- MaxCut-GW é 0.878-aprox. (e não mais !!) (Karloff, 1996)

Problema da Satisfatibilidade Máxima (Max-2SAT)

Problema Max-2SAT

- **Instância:** Cláusulas $\mathcal{C} = \{C_1, \ldots, C_m\}$ de tamanho ≤ 2 sobre variáveis $V = \{x_1, \ldots, x_n\}$. Cada cláusula C_j tem um peso $w_j \geq 0$.
- **Objetivo:** Obter uma valoração das variáveis (0 ou 1) que maximize o peso total das cláusulas satisfeitas.

Formulação de Programação Quadrática Inteira

- Usamos inteiros $y_1, \ldots, y_n$ assoc. às var. e mais y_0 todos em $\{-1, 1\}$
- Interpretação: $x_i = 1$ se $y_i = y_0$; $x_i = 0$ caso contrário
- $v(x_i) = \frac{1}{2}(1 + y_0 y_i)$ e $v(\overline{x_i}) = \frac{1}{2}(1 - y_0 y_i)$
- $v(x_i \vee x_j) = 1 - v(\overline{x_i} \wedge \overline{x_j}) = 1 - v(\overline{x_i})v(\overline{x_j}) = 1 - \frac{1-y_0 y_i}{2} \cdot \frac{1-y_0 y_j}{2}$
- $\Rightarrow v(x_i \vee x_j) = \frac{1+y_0 y_i}{4} + \frac{1+y_0 y_j}{4} + \frac{1-y_i y_j}{4}$
- $\Rightarrow v(x_i \vee \overline{x_j}) = \frac{1+y_0 y_i}{4} + \frac{1-y_0 y_j}{4} + \frac{1+y_i y_j}{4}$
- $\Rightarrow v(\overline{x_i} \vee \overline{x_j}) = \frac{1-y_0 y_i}{4} + \frac{1-y_0 y_j}{4} + \frac{1-y_i y_j}{4}$

Problema da Satisfatibilidade Máxima (Max-2SAT)

Problema Max-2SAT (cláusulas de tam. ≤ 2)

- **Instância:** Cláusulas $\mathcal{C} = \{C_1, \ldots, C_m\}$ sobre variáveis $V = \{x_1, \ldots, x_n\}$. Cada cláusula C_j tem um peso $w_j \geq 0$.
- **Objetivo:** Obter uma valoração das variáveis (0 ou 1) que maximize o peso total das cláusulas satisfeitas.

Formulação de Programação Quadrática Inteira

- **Maximizar** $\sum_{C_j \in \mathcal{C}} w_j \cdot v(C_j)$, **restrito a**:
- Para cada $i \in \{0, 1, \ldots, n\}$: $y_i \in \{-1, 1\}$

Formulação de Programação Quadrática Inteira

- **Maximizar** $\sum_{i<j} \Big[a_{ij}(1 - y_i y_j) + b_{ij}(1 + y_i y_j) \Big]$, **restrito a**:
- Para cada $i \in \{0, 1, \ldots, n\}$: $y_i \in \{-1, 1\}$

- Obtém sol. exata $opt(G, w)$, mas problema NP-Difícil.

Problema da Satisfatibilidade Máxima (Max-2SAT)

Problema Max-2SAT (cláusulas de tam. ≤ 2)

- **Instância:** Cláusulas $\mathcal{C} = \{C_1, \ldots, C_m\}$ sobre variáveis $V = \{x_1, \ldots, x_n\}$. Cada cláusula C_j tem um peso $w_j \geq 0$.
- **Objetivo:** Obter uma valoração das variáveis (0 ou 1) que maximize o peso total das cláusulas satisfeitas.

Formulação de Programação Quadrática Inteira

- **Maximizar** $\sum_{i<j} \Big[a_{ij}(1 - y_i y_j) + b_{ij}(1 + y_i y_j)\Big]$, **restrito a**:
- Para cada $i \in \{0, 1, \ldots, n\}$: $\quad y_i \in \{-1, 1\}$

- Obtém soℓ. exata $opt(G, w)$, mas problema NP-Difícil.

Formulação - Relaxação Vetorial - encontrar vetores $v_0, \ldots, v_n$

- **Maximizar** $\sum_{i<j} \Big[a_{ij}(1 - v_i \cdot v_j) + b_{ij}(1 + v_i \cdot v_j)\Big]$, **restrito a**:
- Para cada $i \in \{0, 1, \ldots, n\}$: $\quad v_i \cdot v_i = 1 \qquad$ (tam. $n+1$)

Algoritmo Max2SAT-GW (Goemans e Williamson, 1995)

Formulação - Relaxação Vetorial - encontrar vetores $v_0, \ldots, v_n$

- **Maximizar** $\sum_{i<j} \Big[a_{ij}(1 - v_i \cdot v_j) + b_{ij}(1 + v_i \cdot v_j)\Big]$, **restrito a**:
- Para cada $i \in \{0, 1, \ldots, n\}$: $\quad v_i \cdot v_i \;=\; 1 \qquad\qquad$ (tam. $n+1$)

Algoritmo Max2SAT-GW(V, $\mathcal{C}$)

1. **sejam** $v_0, v_1, \ldots, v_n$ uma solução ótima da Relaxação vetorial acima
2. **seja** $s \;\leftarrow\; \mathit{RandEsfera}(n+1)$ (vetor aleat. tam. $n+1$ e módulo 1)
3. $A \;\leftarrow\; \{x_i \in V :\; s \cdot v_i > 0\}; \qquad B \leftarrow V - A$
4. **se** $s \cdot v_0 > 0$, **então atribua** V às variáveis em A e F às var. em B
5. **senão atribua** V às variáveis em B e F às var. em A

Teorema: Max2SAT-GW é 0.87856-aprox. probabilístico

Prova: Semelhante à do MaxCut-GW. Próximo slide.

Algoritmo Max2SAT-GW (Goemans e Williamson, 1995)

Teorema: Max2SAT-GW é 0.87856-aprox. probabilístico

Prova: Seja X a soma dos pesos das cláusulas satisfeitas pelo algoritmo.

- $X = \sum_{i<j} \left[a_{ij}(1 - y_i \cdot y_j) + b_{ij}(1 + y_i \cdot y_j) \right]$, onde $y_i = sgn(s \cdot v_i)$
- $X = 2 \sum_{i<j} \left[a_{ij} \cdot \mathbb{1}_{sgn(s \cdot v_i) \neq sgn(s \cdot v_j)} + b_{ij} \cdot \mathbb{1}_{sgn(s \cdot v_i) = sgn(s \cdot v_j)} \right]$
- $\mathbb{E}(X) = 2 \sum_{i<j} \left[a_{ij} \cdot \mathbb{E}(\mathbb{1}_{sgn(s \cdot v_i) \neq sgn(s \cdot v_j)}) + b_{ij} \cdot \mathbb{E}(\mathbb{1}_{sgn(s \cdot v_i) = sgn(s \cdot v_j)}) \right]$
- $\mathbb{E}(X) = 2 \sum_{i<j} \left[a_{ij} \cdot \frac{1}{\pi} arccos(v_i \cdot v_j) + b_{ij} \cdot (1 - \frac{1}{\pi} arccos(v_i \cdot v_j)) \right]$
- $\mathbb{E}(X) \geq \sum_{i<j} \left[a_{ij} \cdot \alpha(1 - v_i \cdot v_j) + b_{ij} \cdot \alpha(1 + v_i \cdot v_j) \right]$, para $\alpha = 0.878$
- $\mathbb{E}(X) \geq \alpha \sum_{i<j} \left[a_{ij} \cdot (1 - v_i \cdot v_j) + b_{ij} \cdot (1 + v_i \cdot v_j) \right] \geq \alpha \cdot opt(V, \mathcal{C})$

Algoritmo Max2SAT-GW (Goemans e Williamson, 1995)

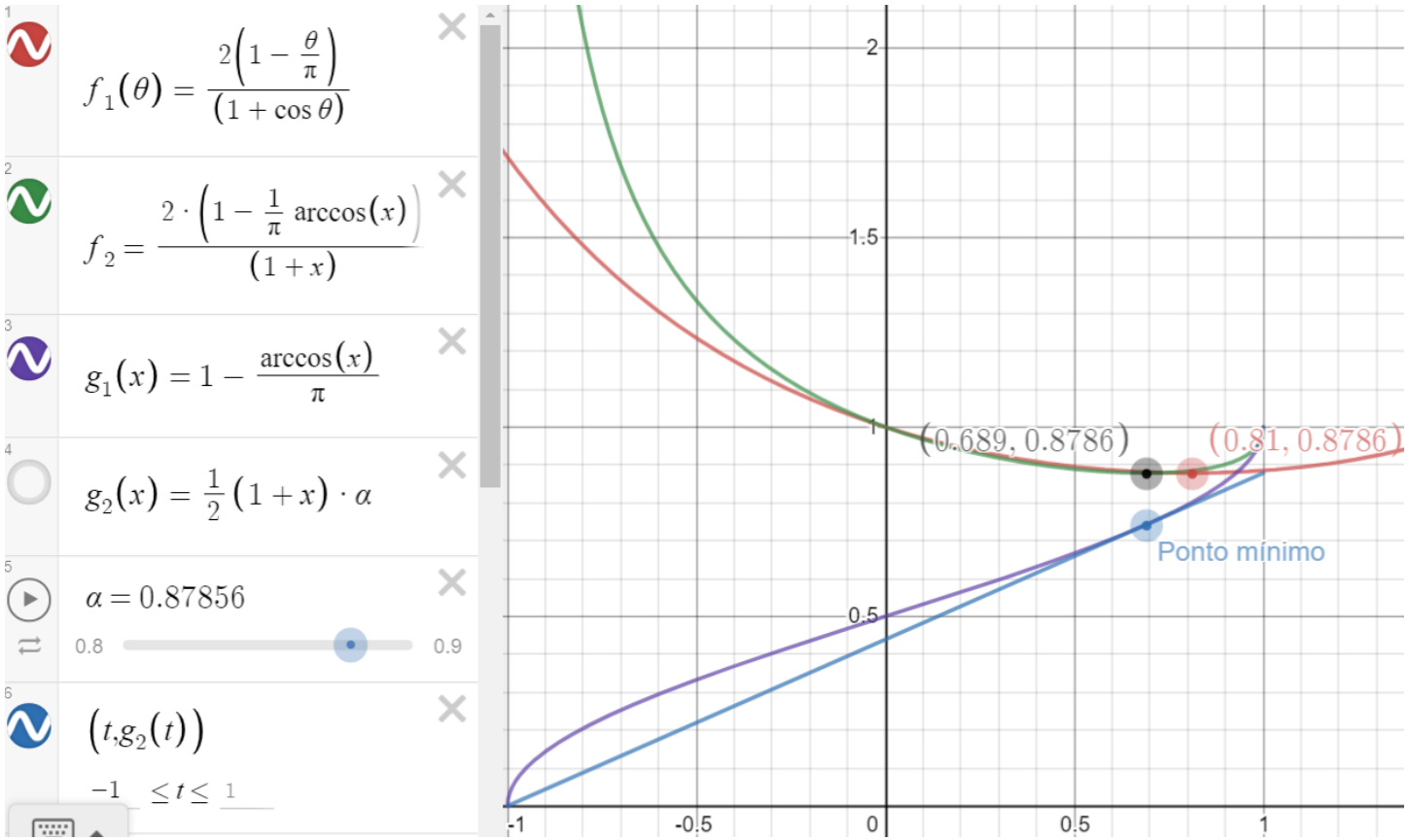


Figure: Gráficos no Desmos: $\alpha = 0.87856$

Conclusão: MaxSAT

Max-2SAT

- 0.5-aprox: Algoritmo MaxSAT-Johnson-Desaleat (Johnson, 1974).
- 0.75-aprox: Algoritmo Max2SAT-Johnson-Desaleat (Johnson, 1974).
- 0.878-aprox: Max2SAT-GW (Goemans, Williamson, 1995)
- 0.931-aprox: Algoritmo Max2SAT-FG (Feige e Goemans, 1995)

MaxSAT

- 0.5-aprox: Algoritmo MaxSAT-Johnson-Desaleat (Johnson, 1974).
- 0.75-aprox: MaxSAT-GW-Combinado (Goemans, Williamson, 1994).
- 0.7584-aprox: MaxSAT-GW-Semidef (Goemans e Williamson, 1995).
- 0.7846-aprox: MaxSAT-AW (Asano, Williamson, 2000)
- 0.7968-aprox: MaxSAT-ABZ (Avidor, Berkovitch, Zwick, 2006)

Algoritmo MaxSAT-GW (Goemans e Williamson, 1995)

- MaxSAT-Johnson: aprox. $1 - \frac{1}{2^k}$ para cláusulas de tam. k.
- MaxSAT-GW-PL: aprox. $1 - (1 - \frac{1}{k})^k$ para cláusulas de tam. k.
- MaxSAT-GW-PQ: aprox. $\alpha = 0.878$ para cláusulas de tam. ≤ 2.

MaxSAT-GW-Full é 0.7555-aprox. (ideia geral)

- Executa o Algoritmo i com probabilidade p_i, onde $p_1 + p_2 + p_3 = 1$

	Tamanho das cláusulas					
	1	2	3	4	5	6
MaxSAT-Johnson	0,5	0,75	0,875	0,9375	0,96875	0,984375
MaxSAT-GW-PL	1	0,75	0,7037037037	0,68359375	0,67232	0,6651020233
MaxSAT-GW-PQ	0,87856	0,87856	0	0	0	0
Fator de aproximação	**0,7555115032**	**0,7555115032**	**0,7555115032**	**0,7757978733**	**0,7853577968**	**0,7893810994**
					p1 = p2 =	0,4785644712
					p3 =	0,04287105758

- Com restrições extras p/ claúsulas tam. ≥ 3, obtem-se aprox. 0.7584