Generalized distances in image segmentation

Krzysztof Chris Ciesielski

Department of Mathematics, West Virginia University
and
MIPG, Department of Radiology, University of Pennsylvania

Part 1: based mainly on two papers with R. Strand, P.K. Saha, and F. Malmberg
Part 2: based on a joint work with J.K. Udupa, A.X. Falcão, and P.A.V. Miranda

90 minutes talk, Special Session on Asymmetric Topology
29th Summer Topology Conference
Staten Island, New York, July 26, 2014

Outline of Part I: distances in image segmentation

1. The problem of image segmentation, by examples
2. Mathematical setting of image segmentation
3. Segmentation via energy minimization and distances
4. Computation of distance functions
5. True TOPOLOGICAL proof of correctness of algorithm A^{appr}_{MBD}
6. Polynomial time algorithm for exact MBD
7. Experiments: comparison of different algorithms for MBD and other distances

Outline of Part I: distances in image segmentation

Example 1 of object segmentation/delineation

Delineation = segmentation of one object and the background

2D image of peppers

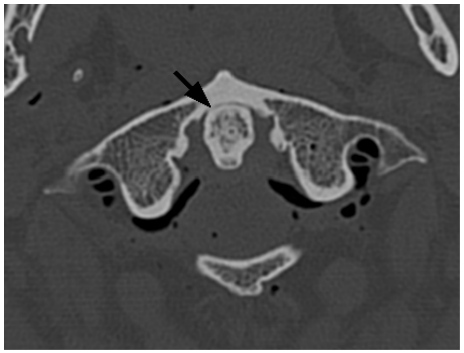
Delineation version 1 (note seeds)

Small changes of parameters can cause big differences:

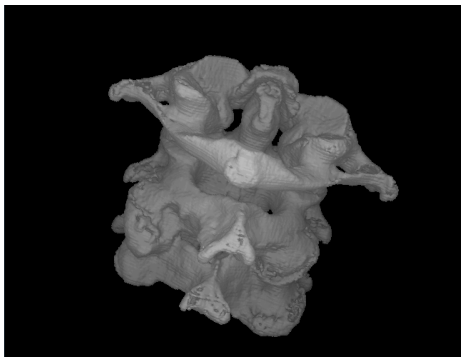
Delineation version 2

Delineation version 3

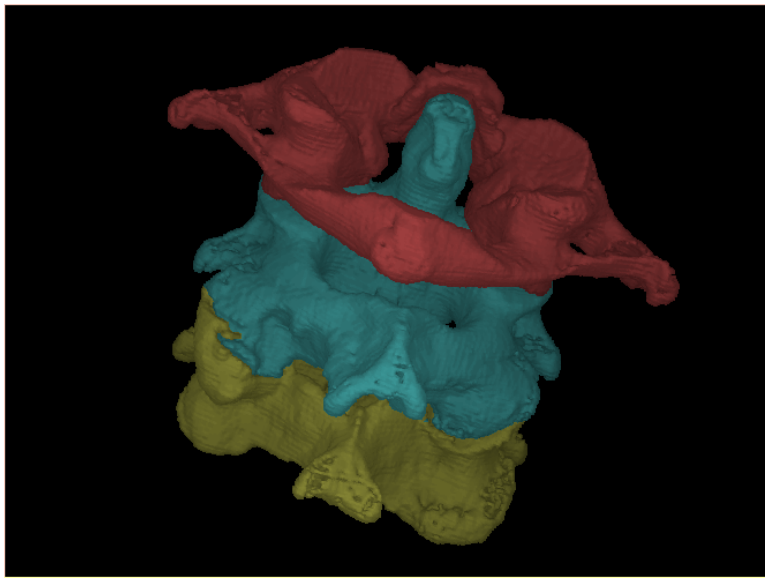
Example 2: a CT image of patient's cervical spine



A slice of an original 3D image

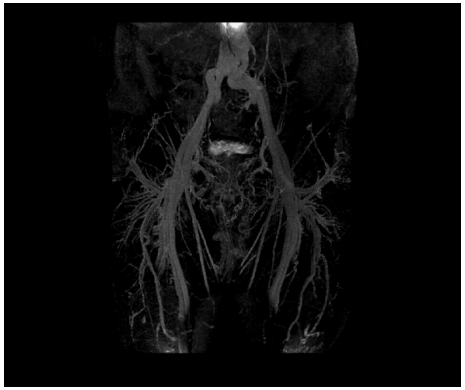


Surface rendition of segmented three vertebrae, together



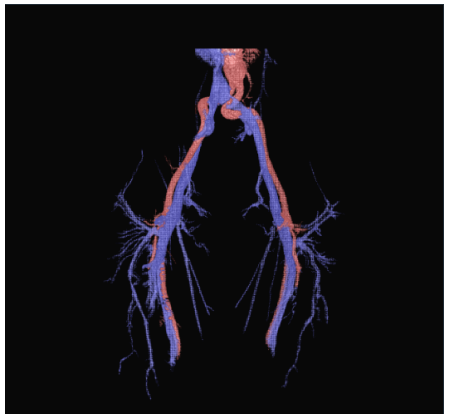
Color surface rendition of the segmented three vertebra

Example 3: An MR angiography image of the body region from belly to knee.



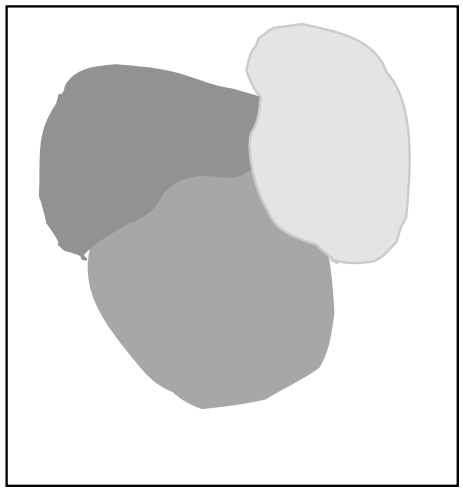
Rendition of an original 3D, contrast enhanced, image

A surface rendition of the entire vascular tree



Color surface rendition of segmented arterial (red) and veinous (blue) trees

Seeds: to help identifying “object of interest”



Which part of this image is “the object?”

Commonly, “an operator” (human or automaton) indicates:

- object via one set, S, of seeds
- background via another set, T, of seeds

General problem of segmentation of images

Find a procedure/algorithm which, given a digital image (of some kind, e.g., 2D or 3D; terrain, medical, or faces; etc) produces its segmentation. The procedure should satisfy

User expectation:

- the resulted segmentations are close to what a user/expert could expect, with as little human interaction as possible (e.g., restricted to indication of the objects with seed sets);

Computational requirement:

- there is an efficient algorithm that can perform the computational part(s) of the procedure.

Goal of this talk: discuss some segmentation algorithms.

Outline of Part I: distances in image segmentation

Digital image as a function

- A nD *digital image* can be identified with a function f from image *scene* C (finite, usually rectangular, subset of $\mathbb{R}^n$) into $\mathbb{R}^k$

$$f : C \to \mathbb{R}^k$$

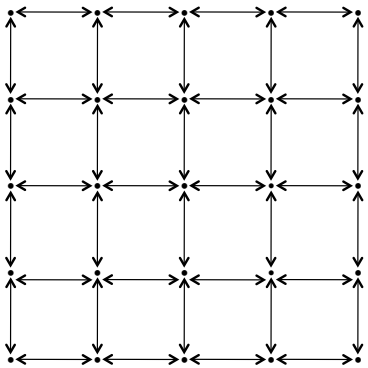
- The elements c of C are pixels (in 2D), voxels (in 3D), or, in general, spels (for *space elements*).

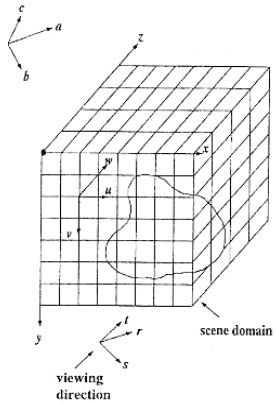
- The value $f(c)$ represents image intensity at c, a k-dimensional vector each component of which indicates a measure of some aspect of the signal, like color.

- Later, we will talk on continuous (idealized) images, defined on open regions Ω in $\mathbb{R}^n$.

Image scene

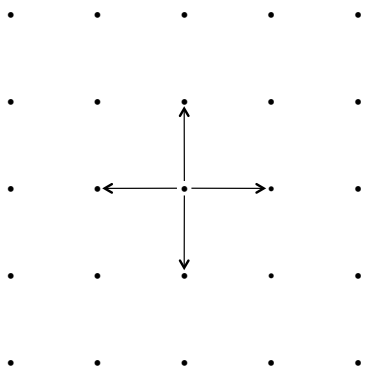
Typically, scene is of rectangular character, as



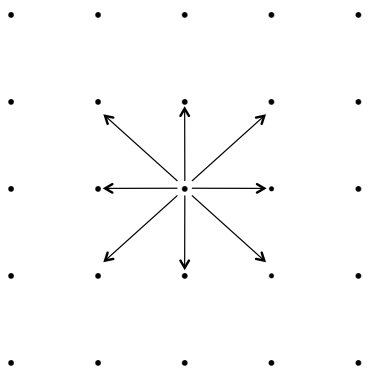


- It comes with a topological/graph structure:
- as a graph $G = \langle C, E \rangle$, *edges* connecting "nearby" spels;
- topologically, such these edges form *adjacency relation*.

Typical edges (2D scenes) and cost functions



edges for 4 adjacency



edges for 8 adjacency

Adjacency relation need not be symmetric; it can be considered as a closure operator (inducing pre-topology).

Information of image is often coded via edge cost/weight function $w(c, d)$ for each edge $\langle c, d \rangle$ (i.e., c adjacent to d).

E.g. proto-distance cost $w(c, d) = \|f(c) - f(d)\|$.

Outline of Part I: distances in image segmentation

Segmentation via energy minimization

Given an image $f\colon C \to \mathbb{R}^k$ and sets $\mathcal{S} = \{S_1, \ldots, S_n\}$ of seeds:

allowable segmentations $\mathbb{P}(\mathcal{S})$ constitute of

the families $\mathcal{P} = \{P_1, \ldots, P_n\}$ of sets with $S_i \subset P_i \subset C$;

usually (not always) sets P_i need to be pairwise disjoint;

in this talk: $\mathcal{P}$ must cover C.

If for any such $\mathcal{P}$ we associate its cost $\varepsilon(\mathcal{P}) \geq 0$

a "good" segmentation is one minimizing an energy ε, i.e.,

$$\arg \min_{\mathcal{P} \in \mathbb{P}(\mathcal{S})} \varepsilon(\mathcal{P})$$

Distance-based energy & Voronoi-like segmentation

Let $d\colon C \times C \to [0,\infty)$ be a generalized distance (i.e., symmetric and satisfying the triangle inequality) associated with an image $f\colon C \to \mathbb{R}^k$.

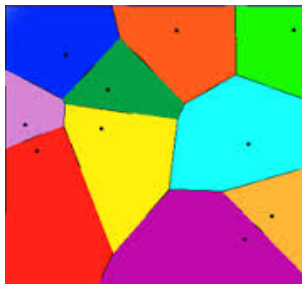
For $\mathcal{P} = \{P_1, \ldots, P_n\}$ from $\mathbb{P}(\mathcal{S})$, $\mathcal{S} = \{S_1, \ldots, S_n\}$, let

$$\varepsilon(x, \mathcal{S}) = \max\{d(x, S_i)\colon x \in P_i\} \quad \text{for any } x \in C, \text{ and put}$$

$$\varepsilon_d(\mathcal{P}) = \sum_{x \in C} \varepsilon(x, \mathcal{S}).$$

A Voronoi diagram (for d and $\mathcal{S}$) is a $\mathcal{P}_\mathcal{S} = \{P_1, \ldots, P_n\} \in \mathbb{P}(\mathcal{S})$, where

$$P_i = \{x \in C\colon d(x, S_i) \le d(x, S_j) \quad \text{for any } j \neq i\}.$$



Example of $\mathcal{P}_\mathcal{S}$

Theorem

$\mathcal{P} \in \mathbb{P}(\mathcal{S})$ minimizes ε_d iff it refines $\mathcal{P}_\mathcal{S}$.

Note on asymmetry

In some cases, the definition: $\mathcal{P}_S = \{P_1, \ldots, P_n\} \in \mathbb{P}(S)$, makes sense also with

$$P_i = \{x \in C \colon d_i(x, S_i) \leq d_j(x, S_j) \quad \text{for any } j \neq i\},$$

where each d_i is a different generalized distance (possibly, even not symmetric).

This works for the Fuzzy Connectedness distance (discussed below) as shown in a 2003 paper of Carvalho, Herman, Kong.

Subject of forthcoming paper of KC, G. Herman, and Y. Kong.

However, for P_i's to be connected, definition is more involved (related to IRFC, unlike $\mathcal{P}_S$, which is basically RFC).

From path strength to generalized distance

Π — all paths $p = \langle c_0, \ldots, c_k \rangle$ in $G = \langle C, E \rangle$, i.e., $\{c_i, c_{i+1}\} \in E$.

$\Pi_{c,d}$ — all paths from $c \in C$ to $d \in C$.

For a fixed path strength map $\lambda \colon \Pi \to [0, \infty)$

a "distance" is $d_\lambda(c, d) = \min\{\lambda(\pi) \colon \pi \in \Pi_{c,d}\}$.

Example. If $w \colon E \to [0, \infty)$ is an edge weight map on G,

with $w(\{c, d\})$ being a (geodesic) distance from c to d,

then d_Σ is the *geodesic metric*, where

$\Sigma(\langle \pi(0), \pi(1), \ldots, \pi(k) \rangle) = \sum_{i=1}^{k} w(\{\pi(i-1), \pi(i)\})$.

Generalized distance: what is needed from λ?

$d\colon C^2 \to [0,\infty)$ is a *generalized distance mappings* if

it is symmetric and satisfies the triangle inequality.

(We allow possibility that $d(c,c) > 0$ for some $c \in C$.)

Theorem

Assume that for every path $\pi = \langle \pi(0), \pi(1), \ldots, \pi(k) \rangle$

(i) $\lambda(\pi) = \lambda(\langle \pi(k), \pi(k-1), \ldots, \pi(0) \rangle)$, *and*

(ii) $\lambda(\pi) \leq \lambda(\langle \pi(0), \ldots, \pi(i) \rangle) + \lambda(\langle \pi(i), \ldots, \pi(k) \rangle)$ *for every* $0 \leq i \leq k$.

Then d_λ *is a generalized distance.*

All maps d_λ we consider (below) are generalized distances.

Generalized distances used in imaging

- Geodesic Distance, d_Σ, including the "Euclidean" Distance
- Fuzzy Connectedness, FC: if μ is FC connectivity strength for affinity $\kappa\colon E \to [0, M]$ and weight $w(e) = M - \kappa(e)$, then $d_\lambda(c, d) = M - \mu(c, d)$, where $\lambda(\langle c_i \rangle) = \max_i w(\{c_{i-1}, c_i\})$
- Watershed: it is $d_{\beta_w^+}$, where $\beta_w^+(\langle c_i \rangle) = \max_i w(c_i)$
- New Minimum Barrier Distance, d_{β_w} to be defined below
- Fuzzy Distance, FD: it is $d_{\hat{\Sigma}}$, where for $w\colon C \to [0, \infty)$ $\hat{w}(c, d) = \frac{w(c)+w(d)}{2}$ and $\hat{\Sigma}(\langle c_i \rangle) = \sum_i \hat{w}(\{c_{i-1}, c_i\})$

For distance d and seed sets $S, T \subset C$ (two objects case) put:

$$P(S, T) = \{c \in C\colon d(c, S) < d(c, T)\}.$$

Then $\mathcal{P}(S, T) = \{P(S, T), C \setminus P(S, T)\}$ minimizes ε_d.

We compare $\mathcal{P}(S, T)$ for d_Σ, FC, MBD, FD.

Definition of the Minimum Barrier Distance, MBD

Let $w\colon C \to [0,\infty)$ be vertex weight map, e.g., $w(c) = \|f(c)\|$.

For a path $p = \langle c_i \rangle \in \Pi$ let $\beta_w(p) = \beta_w^+(p) - \beta_w^-(p)$, where

$\beta_w^+(p) = \max_i w(c_i)$ and $\beta_w^-(p) = \min_i w(c_i)$.

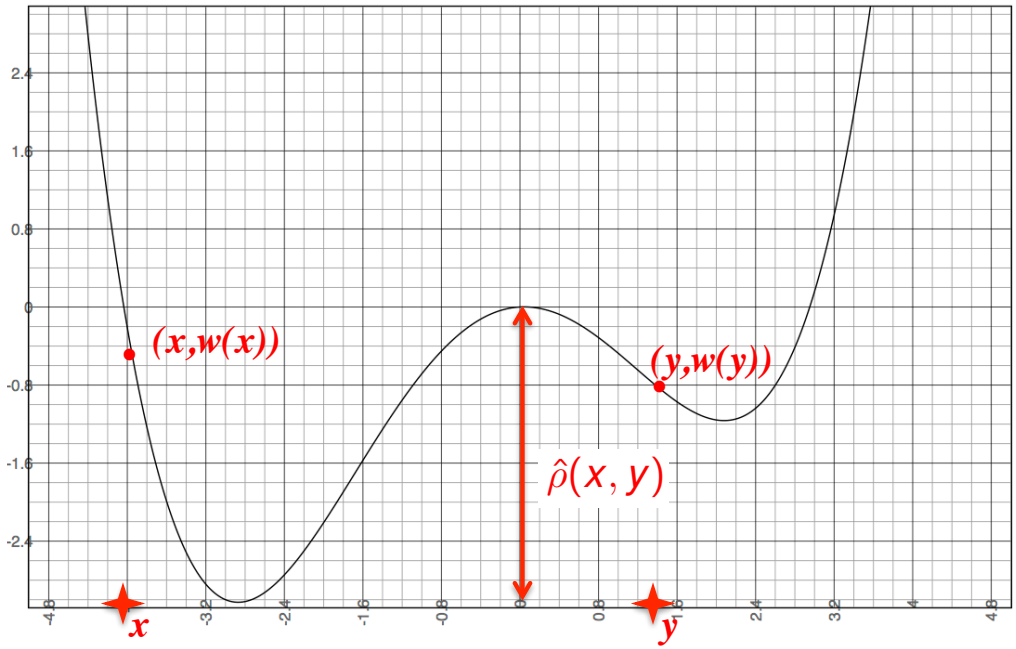
β_w is the barrier cost.

The Minimum Barrier Distance, MBD

between x and y in C

is $d_{\beta_w}(x,y)$, i.e.,

$d_{\beta_w}(x,y) = \min\{\beta_w(p)\colon p \in \Pi_{x,y}\}$.



$\hat{\rho} = d_{\beta_w}$

MBD vs geodesic distance

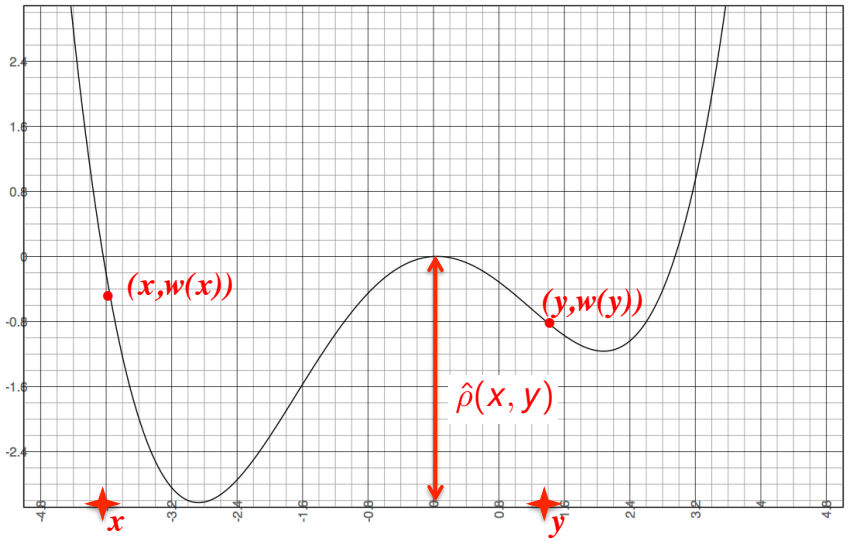
$d_{\beta_w}(x, y) = \min\{c_b(p) : p \text{ is a path in } G \text{ from } x \text{ to } y\}$

$d_{\beta_w}(x, y)$ is, in a way,

a vertical component of

the geodesic distance d_Σ

between x and y.



d_{β_w} is a pseudo-metric: it is symmetric,

satisfies the triangle inequality, and $d_{\beta_w}(x, x) = 0$.

(However, $d_{\beta_w}(x, y)$ can be equal 0 for $x \neq y$.)

Outline of Part I: distances in image segmentation

Standard Dijkstra algorithm, DA, for cost function λ

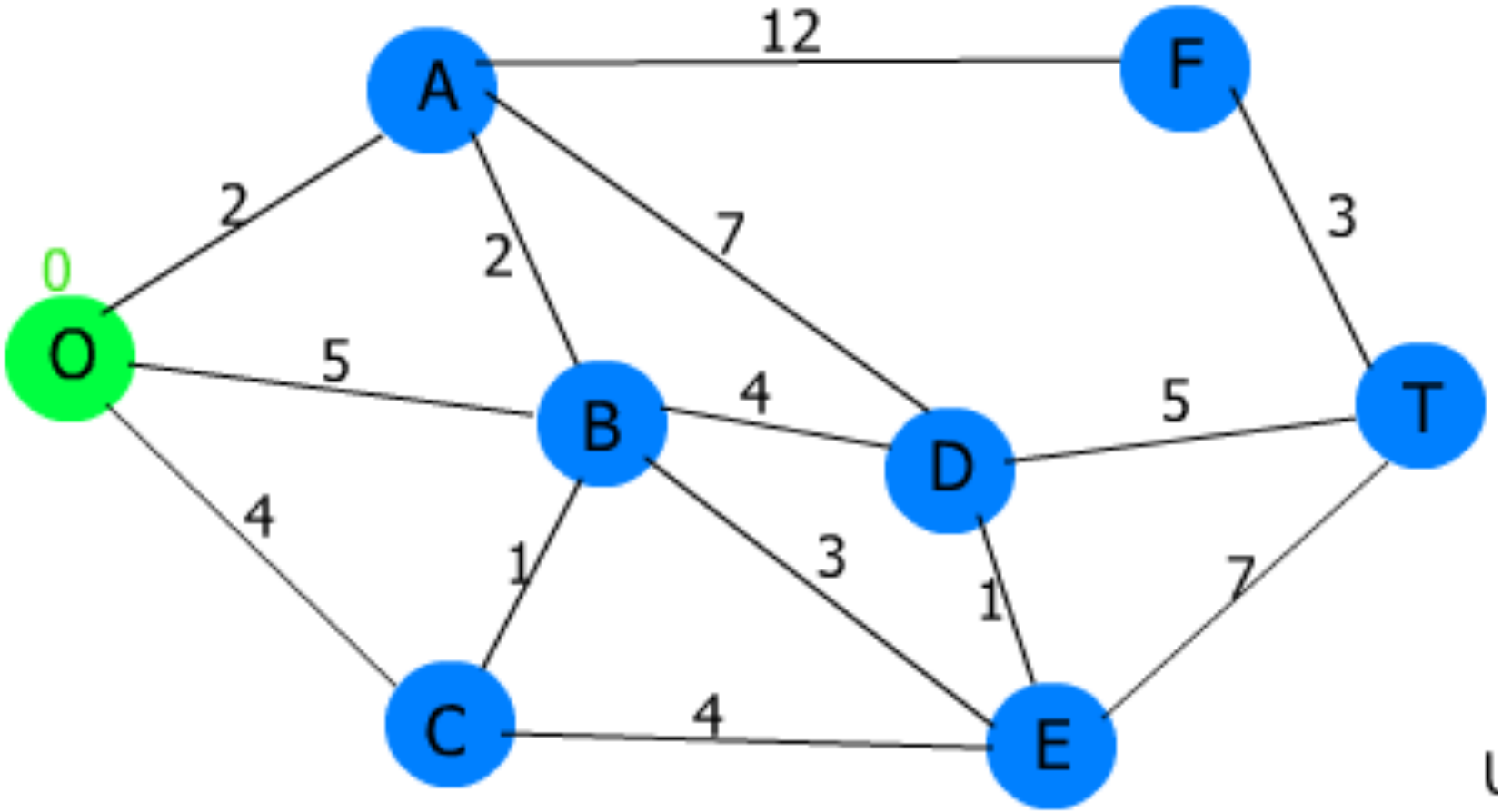
Algorithm 1 Dijkstra (Jarník, Prim) algorithm $DA(\lambda, R)$

Input: Path cost function λ on $G = \langle C, E \rangle$, non-empty $R \subset C$.
Output: For every $c \in C$, a λ-"shortest" path π_c from $r \in R$ to c.
Auxiliary: Queue Q: if c precedes d in Q, then $\lambda(\pi_c) \leq \lambda(\pi_d)$.

1: Init: $p_r = \langle r \rangle$ for $r \in R$, $p_c = \emptyset$ for $c \notin R$, push all $r \in R$ to Q;
2: **while** Q is not empty **do**
3: Pop d from Q;
4: **for** every $c \in C$ connected by an edge to d **do**
5: **if** $\lambda(\pi_d \hat{} c) < \lambda(\pi_c)$ **then**
6: Put $\pi_c = \pi_d \hat{} c$, place c into a proprer place in Q;
7: **end if**
8: **end for**
9: **end while**

Runs in $O(n \ln n)$, where n is the image size.

Dijkstra algorithm example

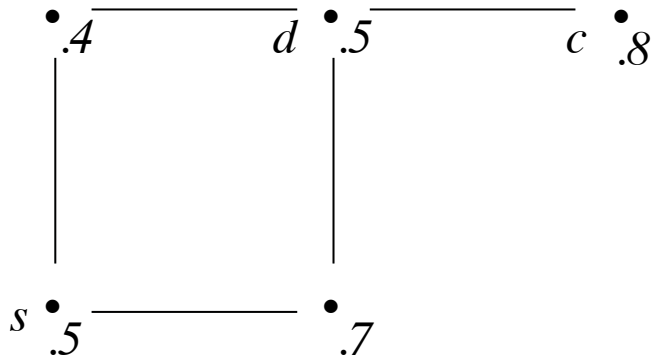


Can Dijkstra Algorithm, DA, find (exact) MBD?

DA returns correctly distances: Geodesic, FC, FD, Watershed,

as their paths strengths are *smooth* in sense of Falcão et al.

DA does not work properly for MBD:



Example: MBD value $d_{\beta_w}(s, c) = .8 - .5$ for the indicated w.

$DA(\beta_w, \{s\})$ returns suboptimal π_c, with $\beta_w(\pi_c) = .8 - .4$.

Fast algorithms approximating MBD

Algorithm 2 Double-Dijkstra $A^{appr}_{MBD}(\{s\})$

Input: A vertex weight map w on a graph $G = \langle C, E \rangle$, an $s \in C$.
Output: A map $\varphi(\cdot, \{s\}))$.
begin

1: Run $DA(\beta^+_w, \{s\}))$; record $d_{\beta^+_w}(c, \{s\})) = \beta^+_w(\pi_c)$ for $c \in C$;
2: Run $DA(\beta^+_v, \{s\}))$, where $v = M - w$ and $M = \max_{c \in C} w(c)$, and record $d_{\beta^-_w}(c, \{s\})) = M - \beta^+_v(\pi_c)$ for every $c \in C$;
3: Return $\varphi(\cdot, \{s\})) = d_{\beta^+_w}(c, \{s\})) - d_{\beta^-_w}(c, \{s\}))$ for $c \in C$;

end

The output of $A^{appr}_{MBD}(\{s\})$ approximates MBD $d_{\beta_w}(\cdot, \{s\}))$:

Outline of Part I: distances in image segmentation

$\varphi(\cdot, \{s\})) \approx$ True MBD $d_{\beta_w}(\cdot, \{s\}))$

$G = \langle C, E, w \rangle$ — graph of a rectangular k-D image f, $w = \|f\|$,

$\varepsilon = \max\{|w(x) - w(y)| : x, y \in C \text{ are } (2^k - 1)\text{-adjacent}\}$.

Theorem ($\varphi(c, s) \leq d_{\beta_w}(c, s) \leq \varphi(c, s) + 2\varepsilon$)

Proof is based on deep result on continuous equivalent of MBD:

For f being continuous on a simple connected domain,

Main Lemma: continuous-$\varphi(c, d)$ = continuous-$d_{\beta_w}(c, d)$.

Proof of Thm:
(1) Extend f to continuous $\hat{f}$ via k-linear interpolation.
(2) Find continuous path $p \in \Pi_{x,y}$ with $\beta_w(p) \approx \varphi(x, y)$.
(3) Digitize p.

continuous-$\varphi(c,d)$ = continuous-$d_{\beta_w}(c,d)$: definitions

Input: Continuous function $f\colon D \to \mathbb{R}$, considered as an image,

where $D = \prod_{i=1}^{k}[a_i, b_i]$ $(a_i, b_i \in \mathbb{R})$.

For a (continuous) path $p\colon [0,1] \to D$ its barrier cost is

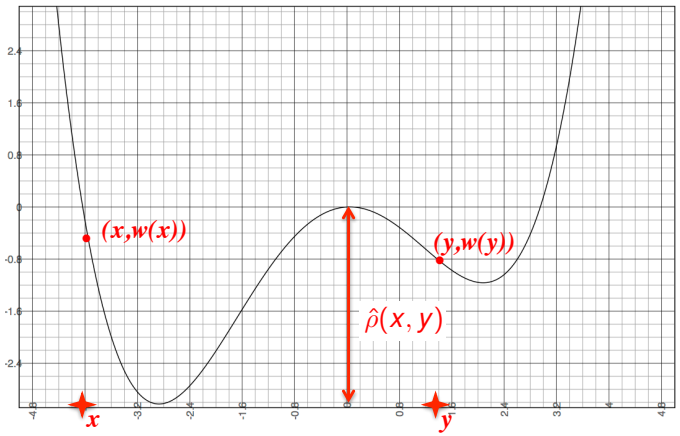
$$c_b(p) = \max_t w(p(t)) - \min_t w(p(t)), \quad \text{here } w = f.$$

(Note that max and min are attained, as $w \circ p$ is continuous.)

The continuous-d_{β_w}, barrier dist. ρ,

between $x, y \in D$ is given by:

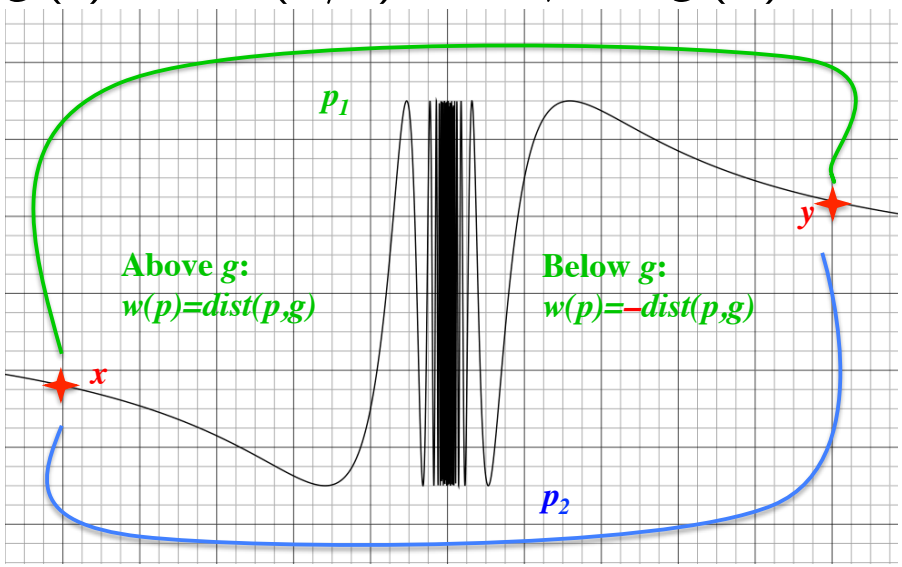
$$\rho(x,y) = \inf\{c_b(p)\colon p \text{ from } x \text{ to } y\}$$



Difficulties: Topologists sine curve example

In $\rho(x, y)$, operation inf cannot be replaced with min:

$g(t) = \sin(1/t)$ for $t \neq 0$, $g(0) = 0$



$$\rho(x, y) = \inf\{c_b(p) : p \in \Pi_{x,y}\}$$

Put $c_{\min}(p) = \min_t w(p(t))$

and $c_{\max}(p) = \max_t w(p(t))$

$$c_{\min}(p_1) = 0 < c_{\max}(p_1)$$

$$c_{\max}(p_2) = 0 > c_{\min}(p_2)$$

For $\varphi(x, y) = \min_{p \in \Pi_{x,y}} c_{\max}(p) - \max_{p \in \Pi_{x,y}} c_{\min}(p)$

$$c_{\max}(p_2) - c_{\min}(p_1) = 0 = \varphi(x, y) = \rho(x, y) < c_b(p)$$

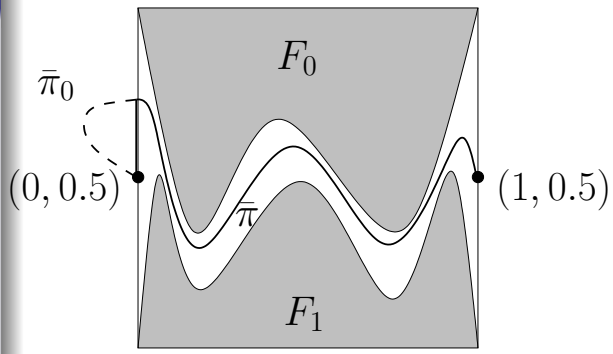
for any $p \in \Pi_{x,y}$.

Proof of: continuous-$\varphi(c,d)$ = continuous-$d_{\beta_w}(c,d)$

Using Alexander's lemma we prove:

Lemma

If $F_0, F_1 \subset [0,1]^2$ *are closed disjoint s.t.* $F_0 \setminus (0,1)^2 \subset (0,1) \times \{1\}$ *and* $F_1 \setminus (0,1)^2 \subset (0,1) \times \{0\}$*, then, there is* $\bar{\pi}\colon [0,1] \to [0,1]^2 \setminus (F_0 \cup F_1)$*, continuous from* $\langle 0, .5\rangle$ *to* $\langle 1, .5\rangle$*.*



Theorem (Non-trivial result on simple connected domains)

If there are $p_1, p_2 \in \Pi_{x,y}$ *with* $a < c_{\min}(p_1)$ *and* $c_{\max}(p_2) < b$*, then there is a single* $p \in \Pi_{x,y}$ *with the range in* (a, b)*.*

Corollary (continuous case)

$\varphi(x,y) = \rho(x,y)$ *for a w on a simple connected domain D.*

Outline of Part I: distances in image segmentation

$A^{appr}_{MBD}(S)$ and $DA(\beta_w, S)$: pros and cons

- Both fast, in order between $O(n)$ and $O(n \ln n)$, $n = |C|$.
- $A^{appr}_{MBD}(S)$ underestimates MBD, with known error rate ε; needs to run "simple" DA $|S|$-many times, slowing for large S.
- $DA(\beta_w, S)$ overestimates MBD with unknown error bound; complexity is (essentially) independent of the size of S;

Conjecture

The error of $DA(\beta_w, S)$ does not exceed 2ε, maybe even ε.

So far, no theoretical proof for this.

Simple algorithm for exact MBD

Algorithm 3 $A^{simple}_{MBD}(S)$

Input: A vertex weight w on $G = \langle C, E \rangle$, non-empty $S \subset C$.
Output: The paths p_c from S to c with $\beta_w(p_c) = d_{\beta_w}(c, S)$.

begin

Init: $U = \max\{w(s) : s \in S\}$ and $p_c = \emptyset$ for every $c \in C$;
Push all numbers from $\{w(c) \leq U : c \in C\}$ to a queue Q;
while Q is not empty **do**
    Pop a from Q, run $DA(\beta^+_v, S)$ with $v = w_a$, return π_c's;
        ($w_a(c) = w(c)$ if $w(c) \geq a$, $w_a(c) = \infty$ otherwise)
    for every $c \in C$ **do**
        if $\beta_v(\pi_c) < \beta_w(p_c)$ **then**
            Put $p_c = \pi_c$;
        end if
    end for
end while

end

Faster algorithm for exact MBD

Algorithm 4 $A_{MBD}(S)$

Auxiliary: β_w^--optimal π_c from S to c; a queue Q: if $c \preceq d$ then $\beta_w^+(\pi_c) < \beta_w^+(\pi_d)$ or $\beta_w^+(\pi_c) = \beta_w^+(\pi_d)$ and $\beta_w^-(\pi_c) > \beta_w^-(\pi_d)$.

begin

1: Init: $p_s = \pi_s = \langle s \rangle$ for $s \in S$ and $p_c = \pi_c = \emptyset$ for $c \in C \setminus S$;
2: Push all $s \in S$ to Q;
3: **while** Q is not empty **do**
4: Pop c from Q;
5: **for** every $d \in C$ connected by an edge to c **do**
6: **if** $\beta_w^-(\pi_c \hat{} d) > \beta_w^-(\pi_d)$ **then**
7: Set $\pi_d \leftarrow \pi_c \hat{} d$ and place d into Q;
8: **if** $\beta_w(\pi_d) < \beta_w(p_d)$ **then**
9: Set $p_d \leftarrow \pi_d$;
10: **end if**
11: **end if**
12: End everything;

Correctness of the algorithms for exact MBD

Theorem

Let n be the size of the graph and m be the size of a fix set Z, containing $W = \{w(c) : c \in C\}$. The algorithm computational complexity is either

(BH) $O(m\, n \ln n)$, *if we use binary heap as Q, or*

(LS) $O(m(n+m))$, *if we use as Q a list structure.*

After $A_{MBD}(S)$ terminates, we indeed have $\beta_w(p_c) = d_w(c, S)$ for all $c \in C$. The same is true for $A^{simple}_{MBD}(S)$.

Proof for $A_{MBD}(S)$ is quite intricate; for $A^{simple}_{MBD}(S)$ is quite easy.

However, $A_{MBD}(S)$ executes the main *while* loop considerably fewer times than $A^{simple}_{MBD}(S)$ does.

Outline of Part I: distances in image segmentation

Step 1: Comparison of different algorithms for MBD

- the exact MBD algorithm $A_{MBD}(S)$;
- the interval algorithm $DA(\beta_w, S)$ overestimating MBD;
- $A^{appr}_{MBD}(S)$ executed ones for each seed point; it underestimates MBD, with an error $\leq 2\varepsilon$;
- $A^{\star appr}_{MBD}(S)$ executed only ones even for multiple seeds.

Experiments were conducted on a computer: HP Proliant ML350 G6 with 2 Intel X5650 6-core processors (2.67Hz) and 104GB memory.

The used 2D images, from the grabcut dataset, came with the true segmentations. Their sizes range from 113032 pixels (for 284×398 image) to 307200 (for 640×480 image).

2D images from the grabcut dataset

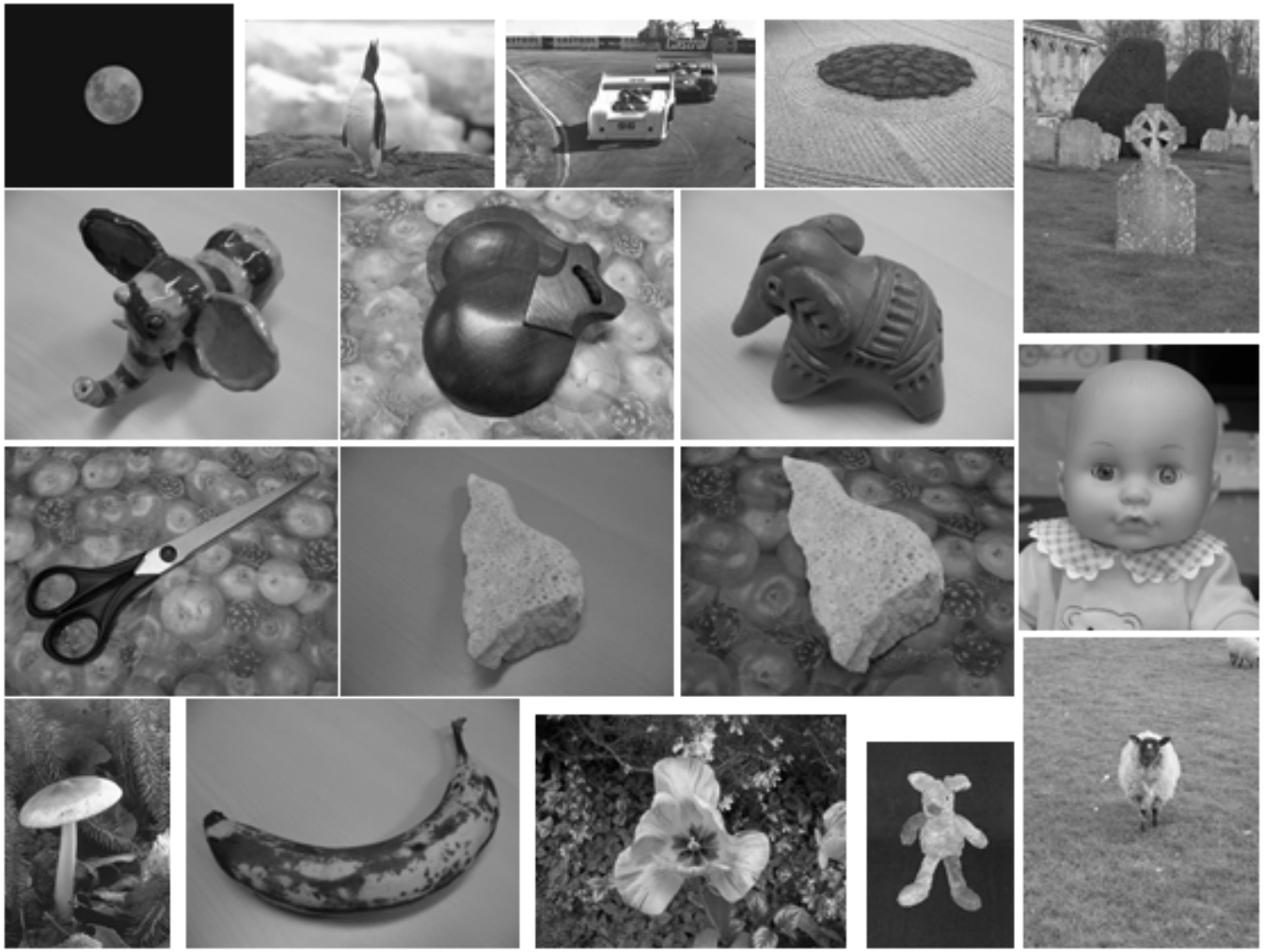
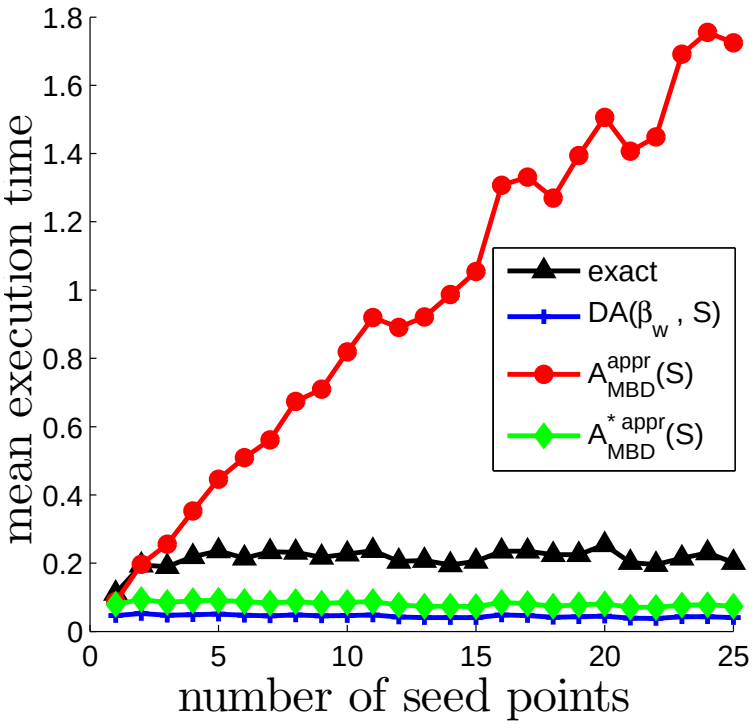
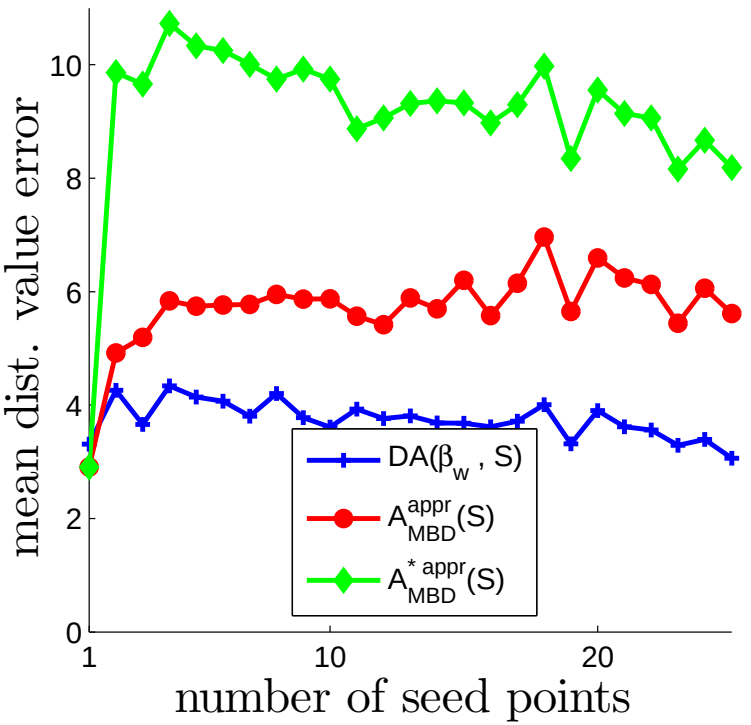


Figure: Images from the grabcut dataset used in the experiments.

Results

For each $s = 1, \ldots, 25$, the following was repeated 100 times:
(1) extract a random image from the database;
(2) generate randomly the set S of s seed points in the image;
(3) run each algorithm on this image with the chosen set S.
Graphs display averages.





More results and conclusions

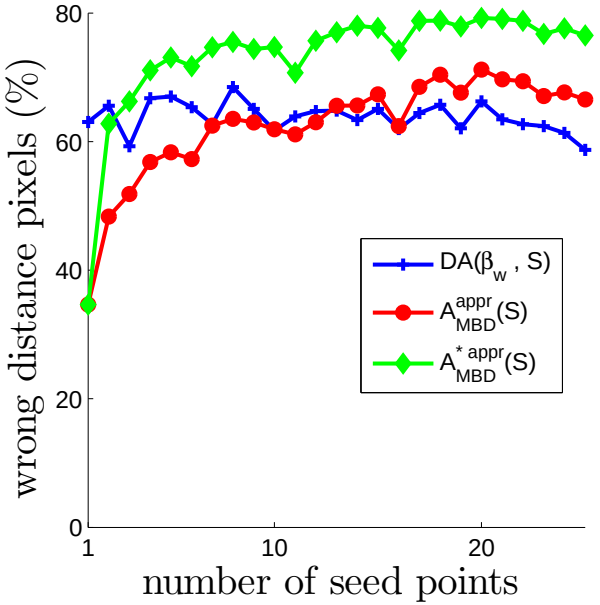


Figure: The mean number pixels with incorrect value of MBD

We declared as "winners," used in the segmentation experiments:

$A_{MBD}(S)$ as it is exact and reasonably fast;

$DA(\beta_w, S)$ as it is the fastest and has the smallest error from approximations.

Step 2: algorithms used in the segmentation valuation

For gray-scale digital images $f : C \to [0, \infty)$:

- The *exact MBD* computed with $A_{MBD}(S)$, where $w(c) = f(c)$.
- An *approximate MBD* computed with $DA(\beta_w, S)$, where $w(c) = f(c)$.
- The *geodesic distance* computed with $DA(\Sigma, S)$, where, for adjacent $c, d \in C$, $w(c, d) = |f(c) - f(d)|$.
- The *fuzzy distance* computed with $DA(\hat{\Sigma}, S)$, where $w(c) = f(c)$.
- The *fuzzy connectedness* computed with $DA(w, S)$, where, for adjacent $c, d \in C$, $w(c, d) = M - \kappa(c, d) = |f(c) - f(d)|$.

We start with the 2D grabcut images.

Speed w.r.t. image size

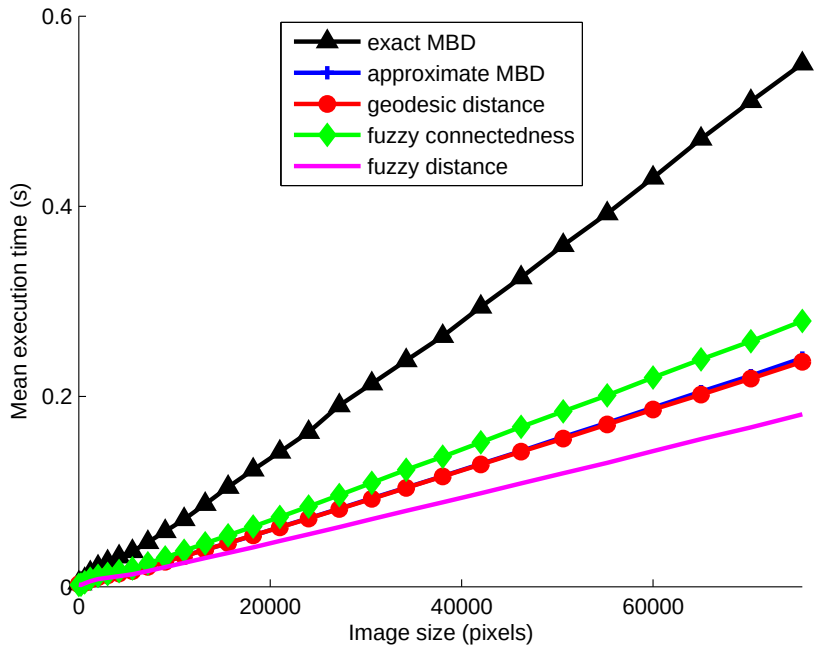
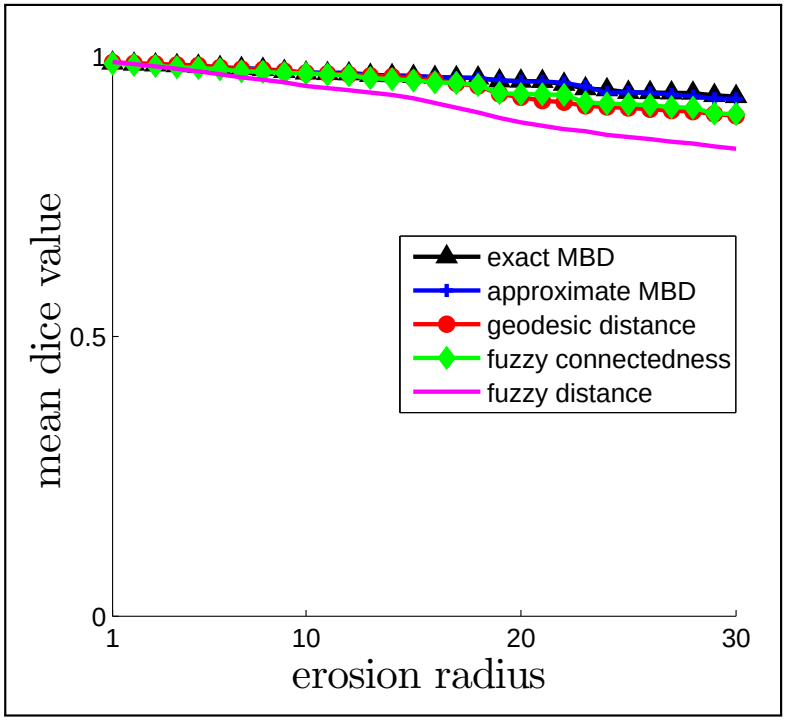


Figure: Mean execution time on small images obtained by cutting out grabcut images. A single seed point is used for each image.

The actual execution time of $A_{MBD}(S)$ depends on the image size in a linear manner, rather than in the (worst case scenario proven) quadratic manner.

Seeds chosen by erosion, no noise or blur



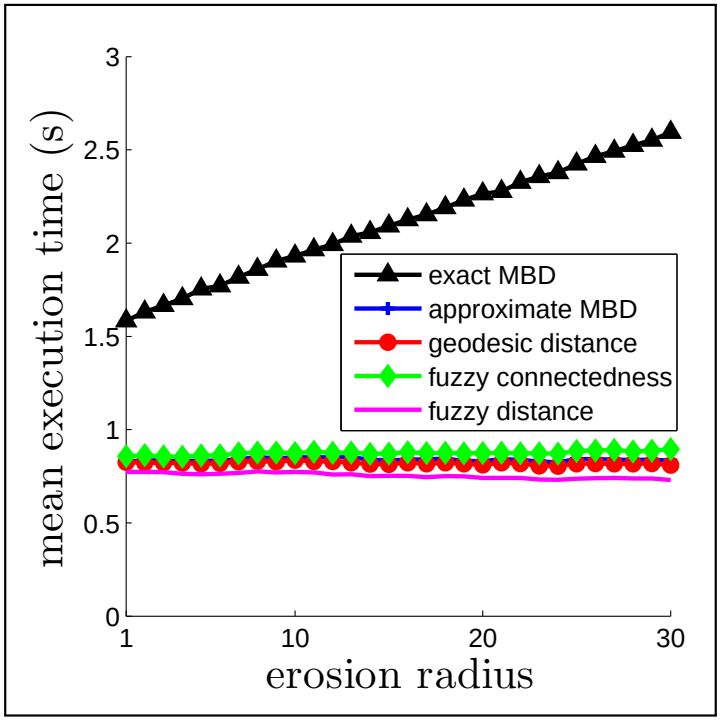


Figure: The value for each algorithm for the seeds chosen for indicated erosion radius represent average over the 17 images.

All algorithms performed well, with just a slight better accuracy for MBD algorithms.

Seeds chosen by the users, no noise or blur

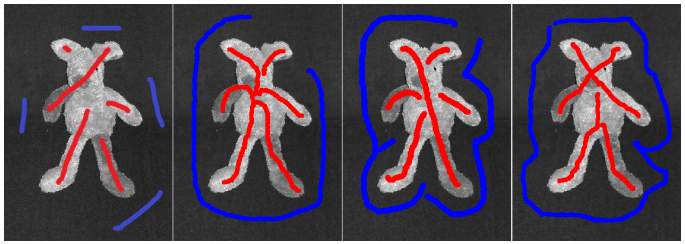


Figure: Example of seed points, users 1–4, respectively.

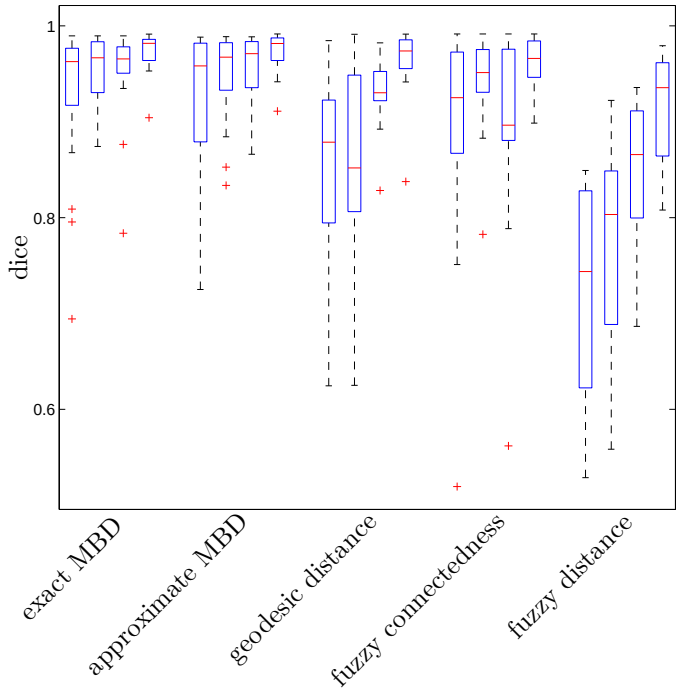
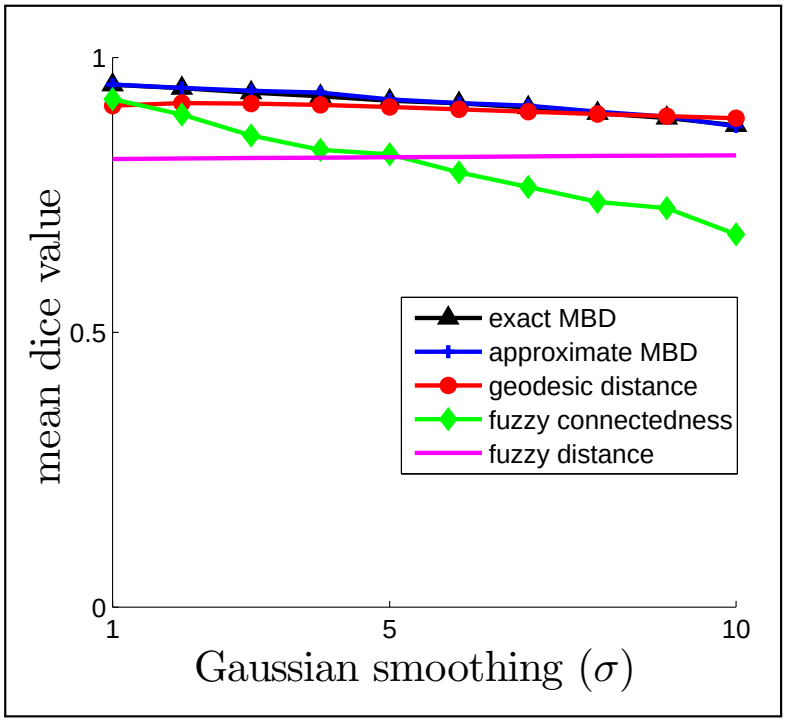


Figure: Boxplots of Dice coefficient, seeds from users 1–4.

Seeds chosen by the users, smoothing added



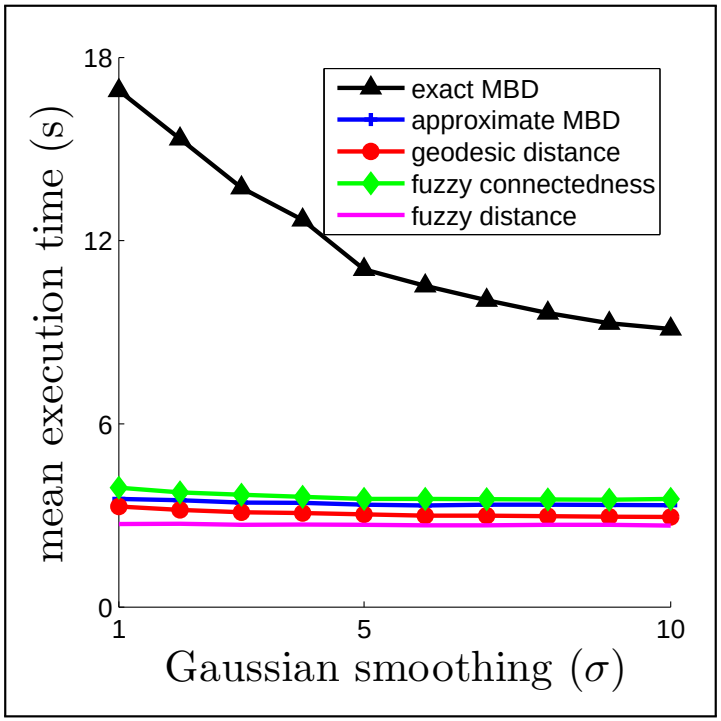
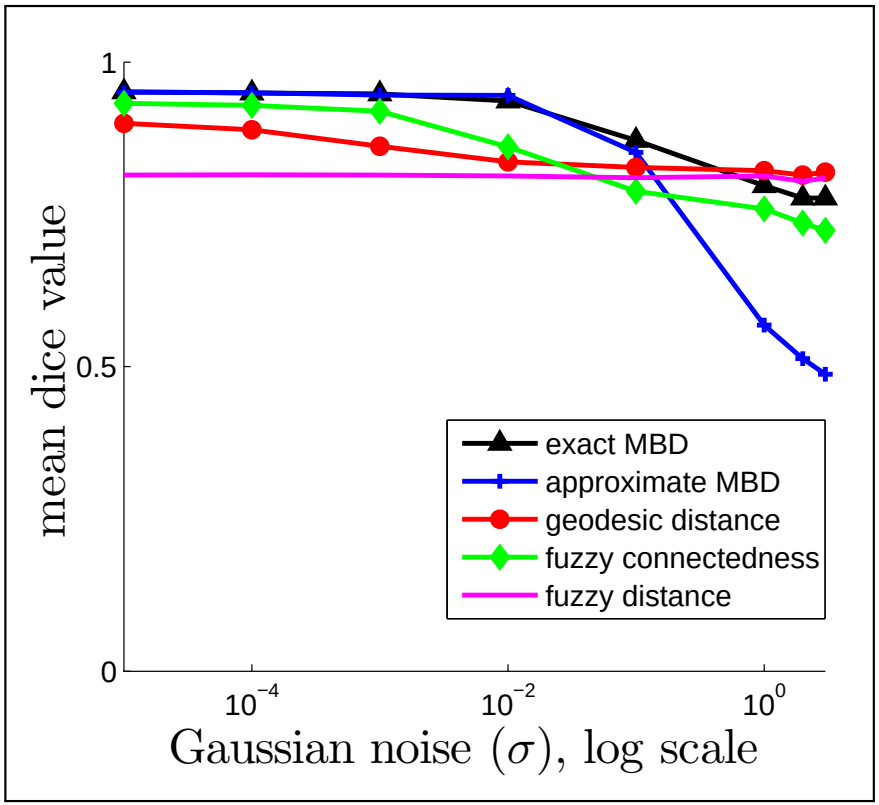


Figure: The performance of the five algorithms as a function of smoothing the images.

MBD algorithms handled smoothing a lot better than FC and FD

Smoothing improves execution time for exact MBD algorithm

Seeds chosen by the users, noise added



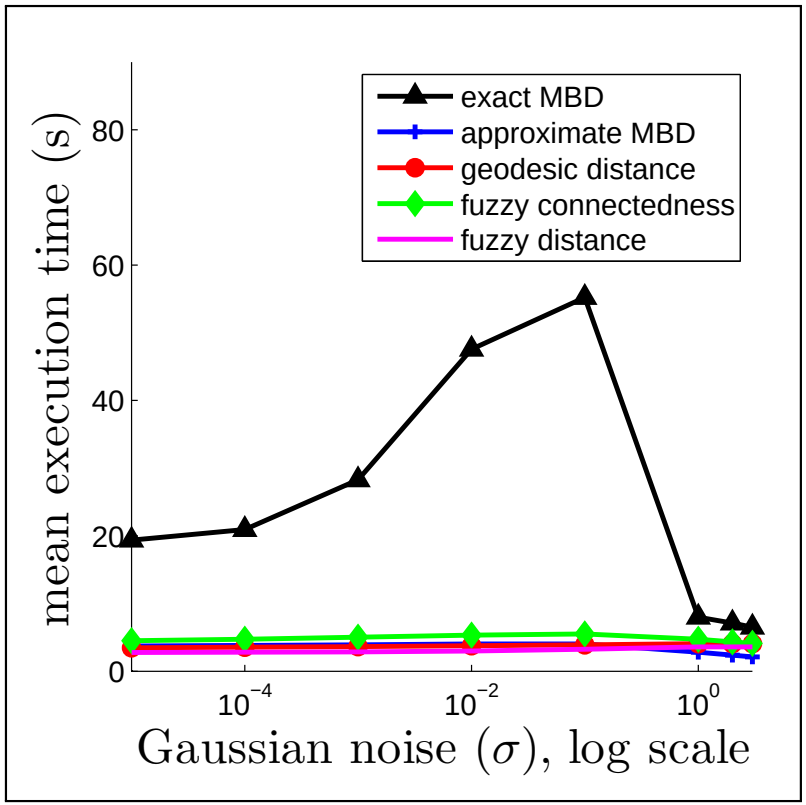


Figure: The performance of the five algorithms as a function of adding noise to the images.

MBD algorithms handled noise better than other algorithms for not very noisy images

Blur added to the images with fixed level of noise

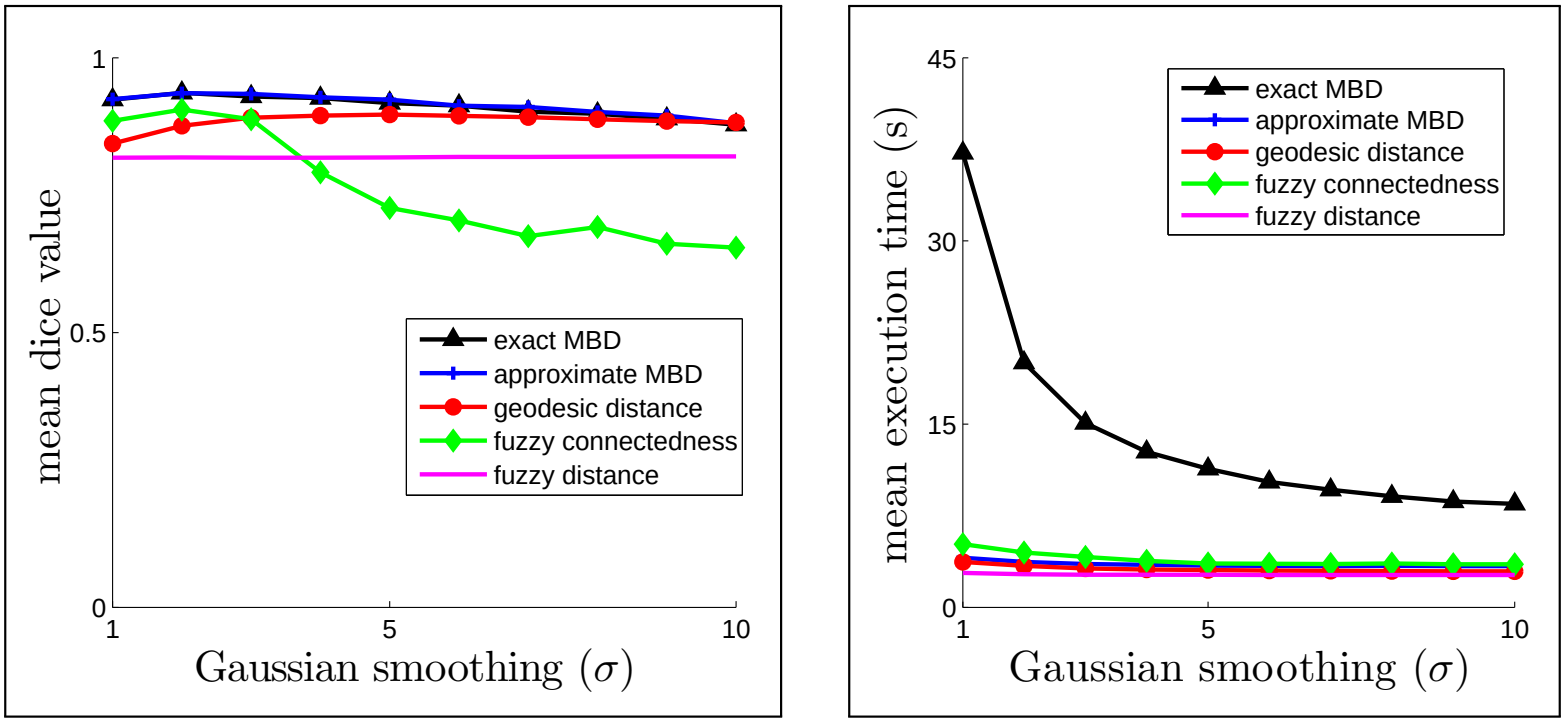
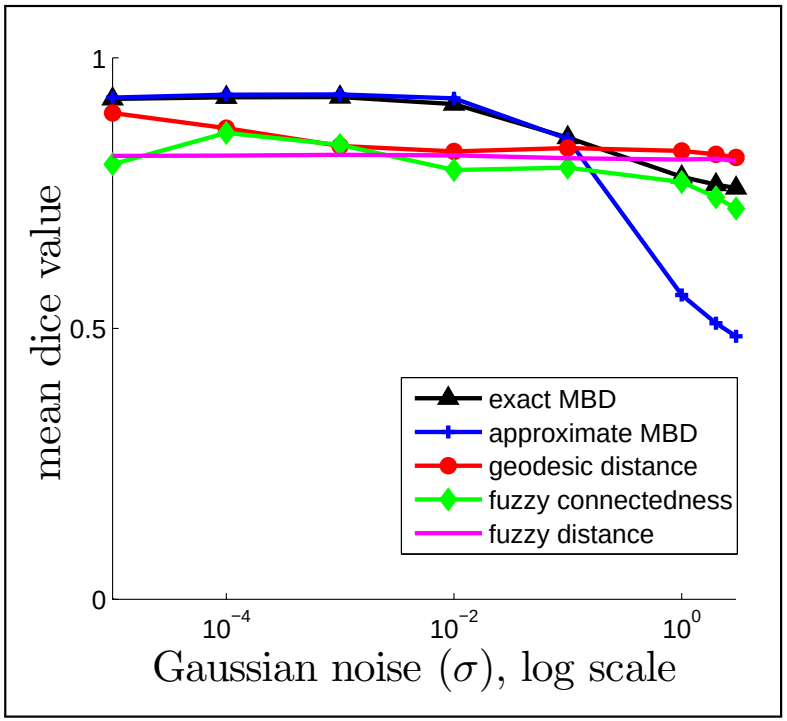


Figure: The performance of the five algorithms as a function of smoothing, applied to the images with added fixed level of noise.

Noise added to the smoothed images



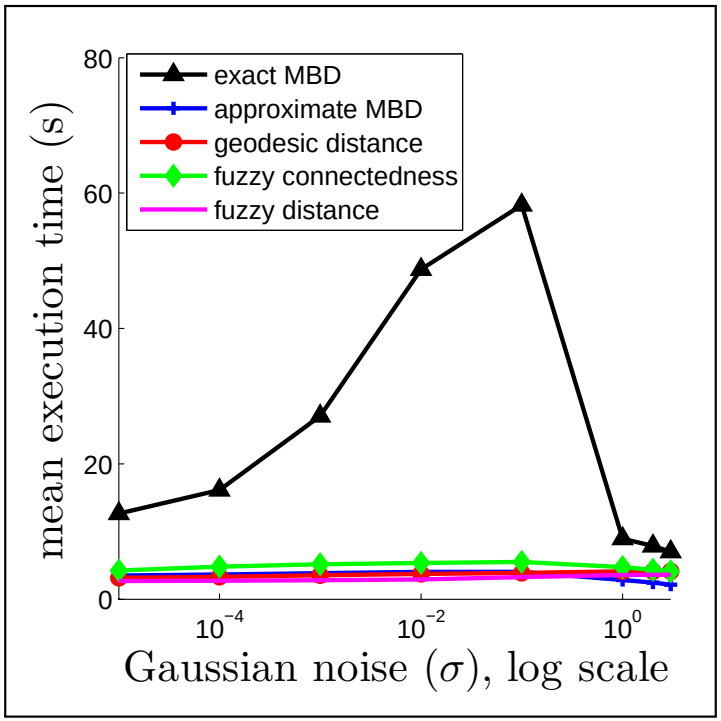


Figure: The performance of the five algorithms as a function of adding noise, applied to the smoothed images.

3D experiments: the image

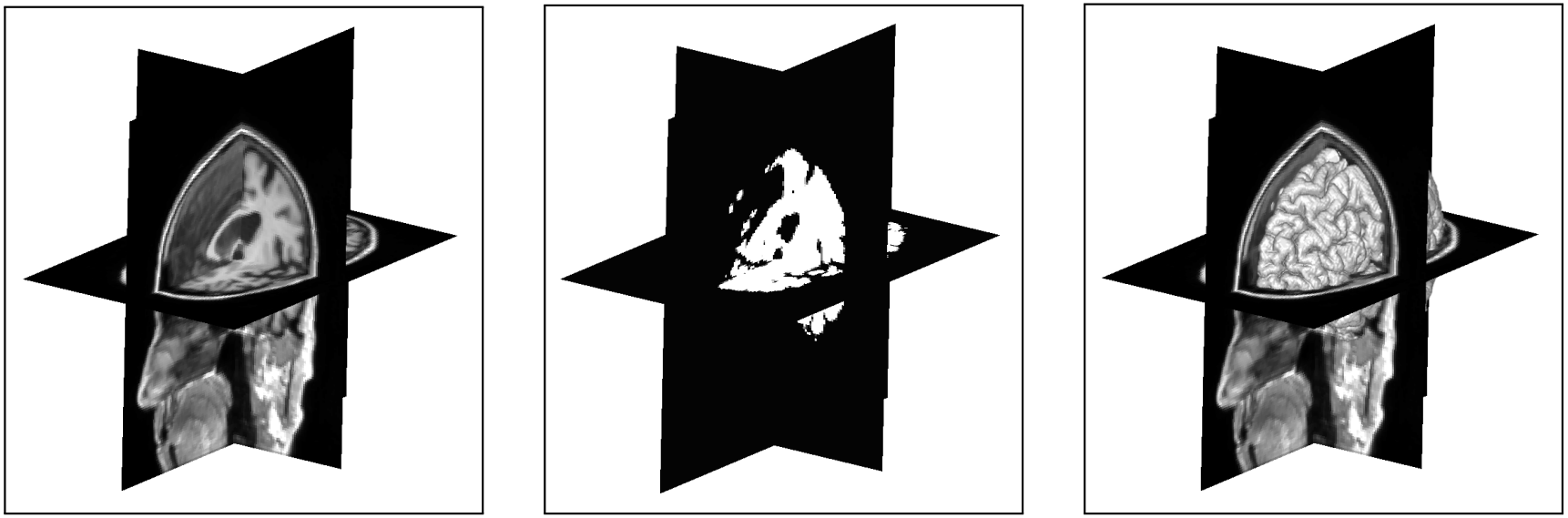


Figure: The 3D T1-weighted MRI image of the brain, smoothed by Gaussian blur with sigma value 0.5. (a) three perpendicular slices; (b) reference segmentation of the same slices; (c) surface rendering of the reference segmentation.

3D experiments: the results

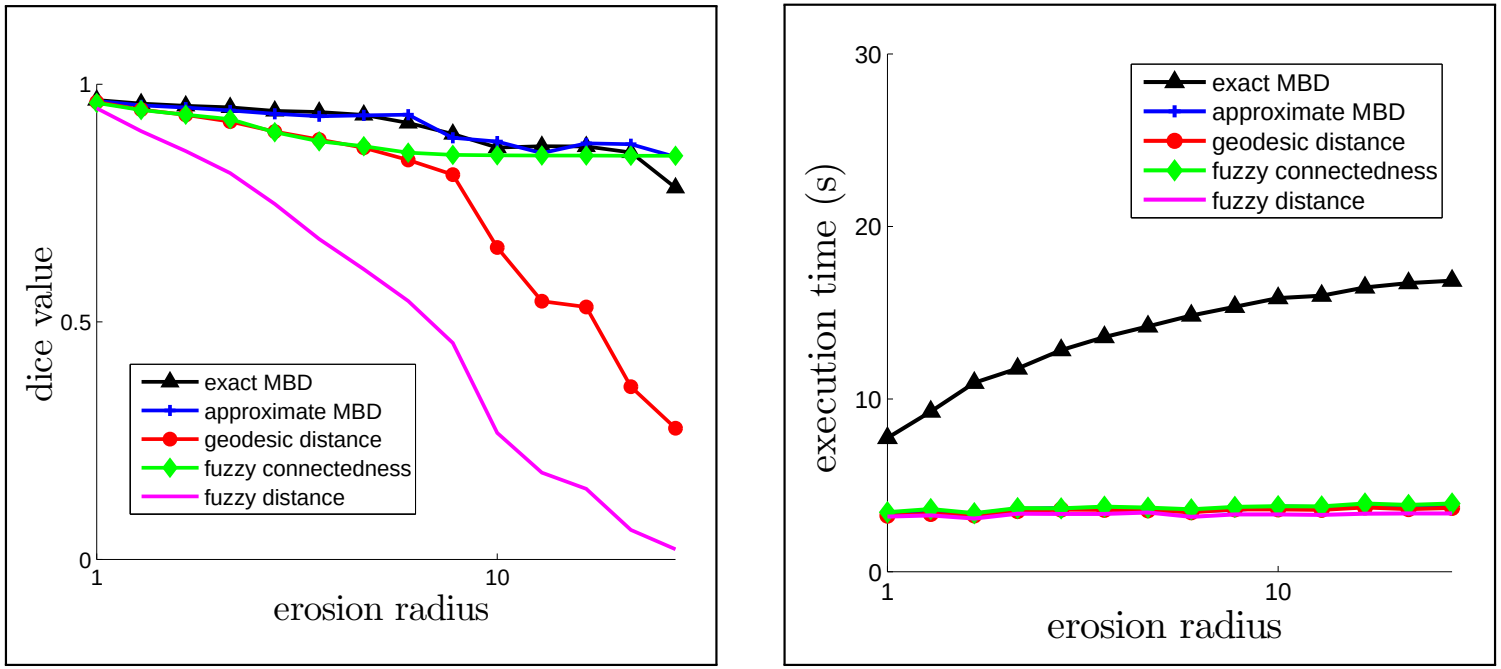


Figure: The performance of the five algorithms on the image for the asymmetrically chosen seeds at the indicated erosion radius.

MBD algorithms compare favorably with the other algorithms

Conclusions

Minimum Barrier Distance:

- Can be efficiently computed: (a) exactly; (b) approximately.
- The segmentations associated with MBD compare favorably with those associates with: geodesic distance (GD), fuzzy distance (FD), and relative fuzzy connectedness (RFC).
- The segmentations associated with MBD are more robust to smoothing and to noise than GD, FD, and RFC.

Part 2: Delineating objects in images via minimization of ℓ_p energies; spanning forests via Dijkstra's and Kruskal's algorithms

Outline of Part 2: Delineating objects via ℓ_p energies

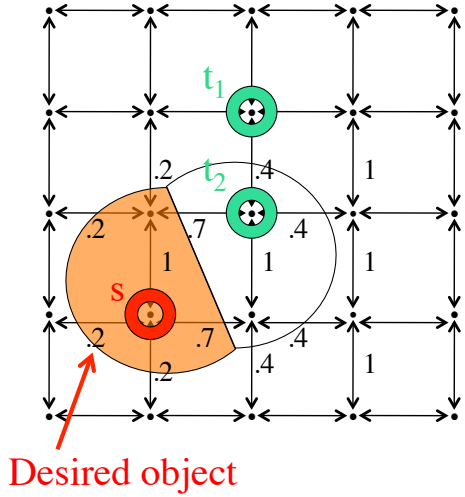
8. ℓ_p distances and related energies
9. Comparison of GC and FC image segmentations
10. Spanning forests, Dijkstra algorithm, IRFC and PW objects
11. Relation between MSF vs OPF: proof

Outline of Part 2: Delineating objects via ℓ_p energies

Heuristic and the definition of boundary

Heuristic: The objects boundary areas should be identifiable in the image, as the areas of sharp image intensity change.

What constitutes boundary $\mathrm{bd}(P)$ of P?



Need graph (or topological) structure $G = \langle V, E \rangle$ on C:

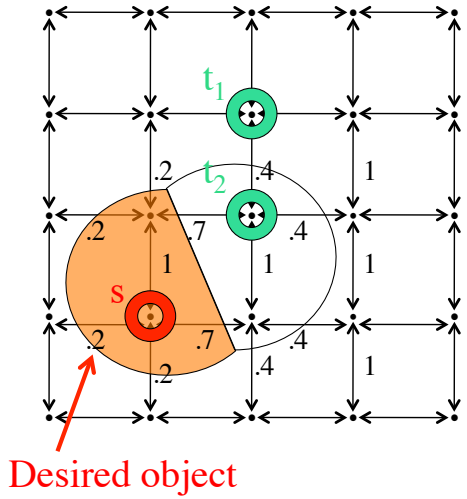
- Pixels $c \in C$ are its vertices, $V = C$;
- Edges $\{c, d\} \in E$ are "nearby" vertices (e.g. as in figure).

$\mathrm{bd}(P)$ is the set of all edges $\{c, d\} \in E$ with $c \in P$ and $d \notin P$

Weighted graphs and ℓ_p cost functions, $1 \leq p \leq \infty$

Assume that with every edge $e = \{c, d\} \in E$ of an image f we have associated its weight/cost $w(e) \geq 0$, which is low, for big $\|f(c) - f(d)\|$.

Typically, $w(e) = e^{-\|f(c)-f(d)\|/\sigma^2}$, see fig.



If $F_P \colon E \to [0, \infty)$, $F_P(e) = w(e)$ for $e \in \mathrm{bd}(P)$ and $F_P(e) = 0$ for $e \notin \mathrm{bd}(P)$, then ℓ_p cost is defined as

$$\varepsilon_p(P) \stackrel{\text{def}}{=} \|F_P\|_p = \begin{cases} \left(\sum_{e \in \mathrm{bd}(P)} w(e)^p\right)^{1/p} & \text{if } p < \infty \\ \max_{e \in \mathrm{bd}(P)} w(e) & \text{if } p = \infty. \end{cases}$$

FC and GC algorithms as minimizers of ε_p

$$\varepsilon_p(P) \stackrel{\text{def}}{=} \|F_P\|_p = \begin{cases} \left(\sum_{e \in \text{bd}(P)} w(e)^p\right)^{1/p} & \text{if } p < \infty \\ \max_{e \in \text{bd}(P)} w(e) & \text{if } p = \infty. \end{cases}$$

$p = 1$: $\varepsilon_1(P) = \sum_{e \in \text{bd}(P)} w(e)$; algorithm admits asymmetric cost

Optimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm optimizes ε_1.

$p = \infty$: $\varepsilon_\infty(P) = \max_{e \in \text{bd}(P)} w(e)$;
Optimization solved by (versions of) Dijkstra algorithm.

ε_∞ optimized objects are returned by the algorithms: Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC, and Power Watershed, PW [C. Couprie *et al*, 2011].

$p = 2$: related to Random Walker, RW, algorithm [Grady, 2006], see next slides.

Fuzzy sets

A map $x\colon C \to [0,1]$ (i.e., $x \in [0,1]^C$) can be considered as a *fuzzy set*, with $x(c)$ giving the degree of membership of c in it.

A hard set $P \subset C$ is identified with a fuzzy set (binary image) $\chi_P \in \{0,1\}^C \subset [0,1]^C$, $\chi_P(c) = 1$ iff $c \in P$.

For $x \in [0,1]^C$ let $\hat{\varepsilon}_p(x) = \|F_x\|_p$, where $F_x\colon E \to [0,\infty)$,

$F_x(\{c,d\}) = |x(c) - x(d)|w(\{c,d\})$ for $\{c,d\} \in E$.

Then $\varepsilon_p(P) = \hat{\varepsilon}_p(\chi_P)$. We can minimize $\hat{\varepsilon}_p$ on

$\hat{\mathcal{P}}(S,T) = \{x\colon x(c) = 1 \text{ for } c \in S \;\&\; x(c) = 0 \text{ for } c \in T\}$

instead of ε_p on $\mathcal{P}(S,T) = \hat{\mathcal{P}}(S,T) \cap \{0,1\}^C$.

Random Walker, RW, algorithm

- RW finds (the unique) $\hat{\varepsilon}_2$ minimizer on $\hat{\mathcal{P}}(S, T)$.
- Defines its output as $P = \{c\colon x(c) \geq .5\}$.

Problems with RW:

1. Output need not be connected (even when S and T are).
2. P need not minimize ε_2 on $\mathcal{P}(S, T)$.

Neither of this happens for ε_1 (i.e. GC) or ε_∞ (i.e. RFC or PW):

Thm: For $p \in \{1, \infty\}$, any minimizer of $\hat{\varepsilon}_p$ on $\hat{\mathcal{P}}(S, T)$ actually belongs to $\mathcal{P}(S, T)$.

(Non)-uniquness of the minimizers for ε_1 and ε_∞

Let $\mathcal{P}_p(S,T) = \{P \in \mathcal{P}(S,T) \colon P \text{ minimizes } \varepsilon_p \text{ on } \mathcal{P}(S,T)\}$.

Both $\mathcal{P}_1(S,T)$ and $\mathcal{P}_\infty(S,T)$ may have more than one element.

However, the outputs of the standard versions of the algorithms:

- GC, from $\mathcal{P}_1(S,T)$,
- RFC, from $\mathcal{P}_\infty(S,T)$, and
- IRFC, from $\mathcal{P}_\infty(S,T)$

are unique in the sense of the next theorem.

GC & FC segmentations — comparison theorem 1

Theorem (Argument minimality)

For $p \in \{1, \infty\}$, $\mathcal{P}_\varepsilon(S, T)$ contains the $\subset$-smallest object.

- *GC algorithm returns the smallest set in $\mathcal{P}_1(S, T)$.*
- *RFC algorithm returns the smallest set in $\mathcal{P}_\infty(S, T)$.*
- *IRFC algorithm returns the smallest set in a refinement $\mathcal{P}^*_\infty(S, T)$ of $\mathcal{P}_\infty(S, T)$.*

Moreover, if n is the size of the image (scene), then

- *GC runs in time of order $O(n^3)$ (the best known algorithm) or $O(n^{2.5})$ (the fastest currently known algorithm)*
- *Both RFC and IRFC run in time of order $O(n)$ (for standard medical images — the intensity range size not too big) or $O(n \ln n)$ (the worst case scenario)*

GC & FC — asymptotic equivalence

Theorem (Asymptotic equivalence of GC and FC)

Let $\mathcal{P}_p^m(S,T)$ be the family $\mathcal{P}_p(S,T)$ for the edge weight function w replaced by its m-th power w^m. Then

- $\mathcal{P}_\infty^m(S,T) = \mathcal{P}_\infty(S,T)$ *and similarly for IRFC algorithm.*

  So, the outputs of RFC and IRFC are unchanged by m.

- $\mathcal{P}_1^m(S,T) \subseteq \mathcal{P}_\infty(S,T)$ *for m large enough.*

In particular, if $\mathcal{P}_\infty(S,T)$ has only one element, then

the output of GC coincides with the outputs of RFC and IRFC

for m large enough.

Outline of Part 2: Delineating objects via ℓ_p energies

Advantages of FC over GC — theoretical angle

Speed: FC algorithms run a lot faster than GC algorithms: $O(n)$ (or $O(n \ln n)$) versus $O(n^3)$ (or $O(n^{2.5})$).

Robustness: RFC & IRFC are unaffected by small seed changes. GC is sensitive for even small seed changes.

Shrinking: GC chooses objects with small size boundary (often with edges with high weights); No such problem for RFC & IRFC

Multiple objects: FC framework handles easily the segmentation of multiple objects, same running time and robustness. GC in such setting leads to NP-hard problem, so (for precise delineation) it runs in exponential time

Iterative approach: RFC has an iterative approach refinement; No such refinement methods exist for GC at present.

Advantages of GC over FC

Boundary smoothness: GC chooses small boudary, so it naturally smooths it; in many (but not all) medically important delineations, this is a desirable feature.

Basic FC framework has no boundary smoothing; if desirable, smoothing requires post processing

Combining image homogeneity info with known object intensity: GC naturally combines information on image homogeneity (binary relation on voxels) with information on expected object intensity (unary relation on voxels);

Combining such informations is difficult to achieve in the FC framework.

Setup of experiments:

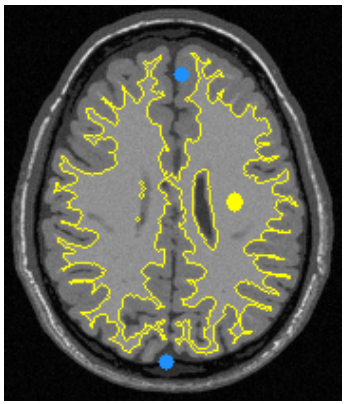
- In each experiment we used 20 MR BrainWeb phantom images (simulated T1 acquisition); graphs show averages.
- Sets of seeds were generated, from known true binary segmentations, by applying erosion operation: the bigger erosion radius, the smaller the seed sets.
- The weight map $w(c, d)$, same for FC and GC, was defined from the image intensity function f as $w(c, d) = -|G(f(c)) - G(f(d))|$, where G is an appropriate Gaussian.

Setup of experiments:

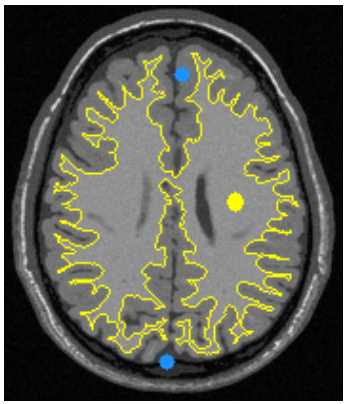
Data parameters: the simulated T1 acquisition were as follows: spoiled FLASH sequence with TR=22ms and TE=9.2ms, flip angle = 30°, voxel size $= 1 \times 1 \times 1\text{mm}^3$, noise = 3%, and background non-uniformity = 20%.

Computer: Experiments were run on PC with an AMD Athlon 64 X2 Dual-Core Processor TK-57, 1.9 GHz, 2×256 KB L2 cache, and 2 GB DDR2 of RAM.

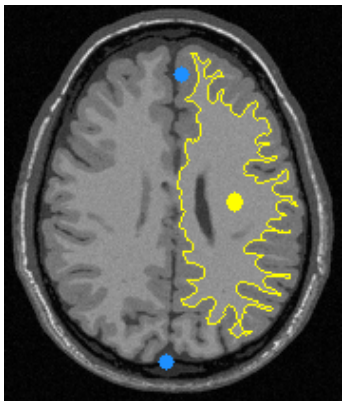
Robustness & shrinking for FC & GC: White Matter



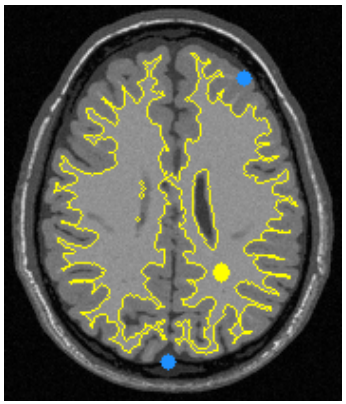
(a) RFC



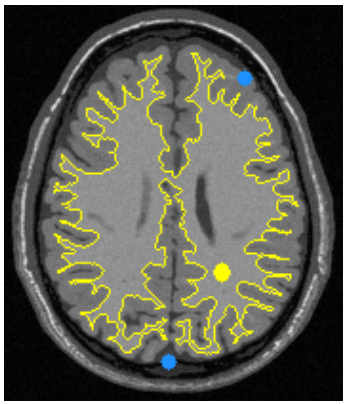
(b) IRFC



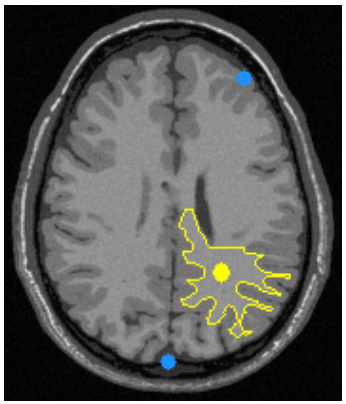
(c) GC



(d) RFC



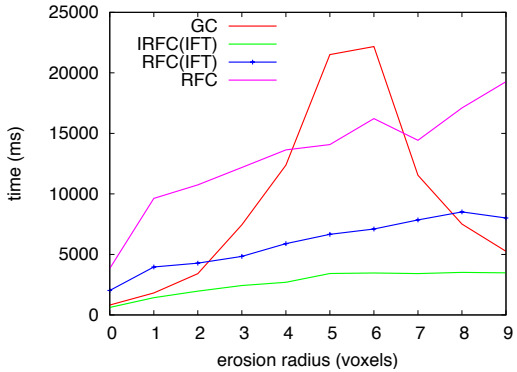
(e) IRFC



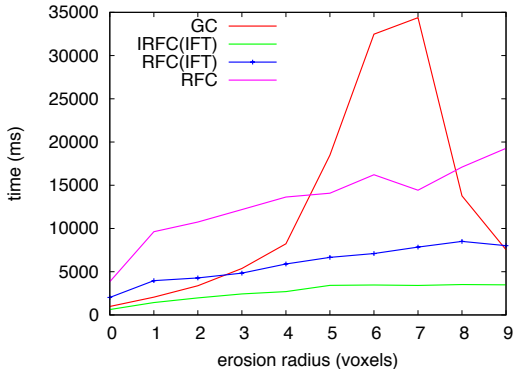
(f) GC

Figure: (a)&(d) and (b)&(e): same outputs for different seeds; (c)&(f) GC: dramatic change of output; seeds choice same as in the FC case

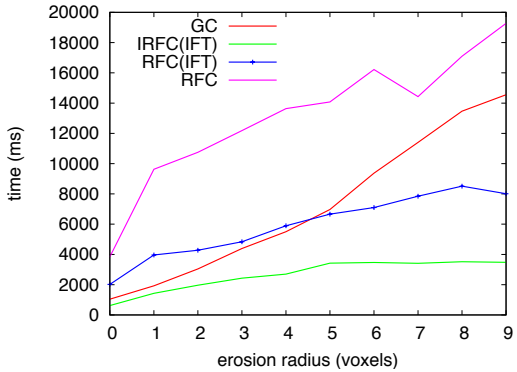
Time & accuracy of FC & GC: segmentation of WM



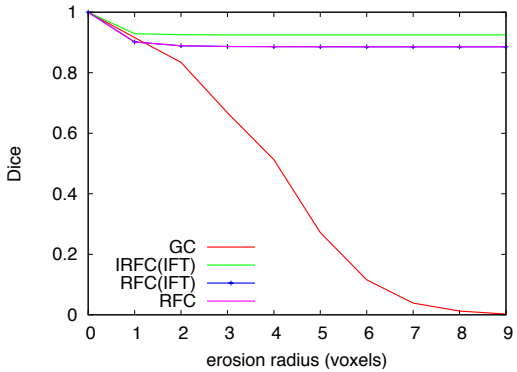
(a) Time for w^1



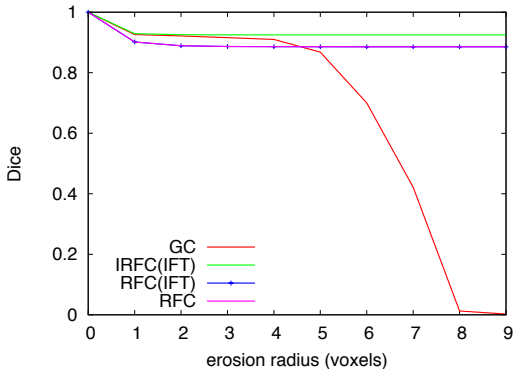
(b) Time for w^5



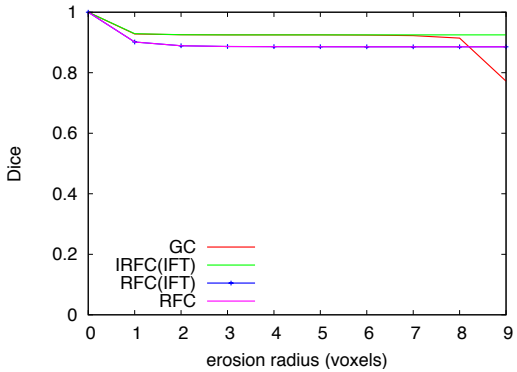
(c) Time for w^{30}



(d) Accuracy for w^1



(e) Accuracy for w^5



(f) Accuracy for w^{30}

FC vs GC: Conclusions

- FC and GC quite similar,
  yet FC has many advantages over GC:
  - FC runs considerably faster than GC
  - FC is robust (seed), while GC has shrinkage problem
  - FC, unlike GC, easily handles multiple-object segmentation
- unless the application requires, in an essential way, the simultaneous use of
  - homogeneity (binary) info on image intensity;
  - expected object intensity (unary) info on image intensity;

  it makes sense to use FC (more precisely IRFC) segmentation algorithm, rather than GC algorithm

Outline of Part 2: Delineating objects via ℓ_p energies

Forests: the powerhouse behind Dijkstra algorithm

Fix weighted graph $G = \langle C, E, w \rangle$ and $\emptyset \neq W \subset C$.

Definition (Spanning Forest w.r.t. W)

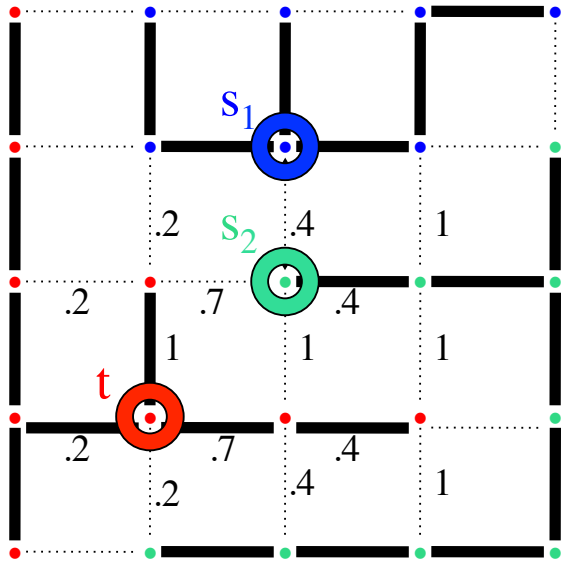
A *forest* for G is any subgraph $\mathbb{F} = \langle C, E' \rangle$ of G free of cycles. $\mathbb{F} = \langle C, E' \rangle$ is *spanning with respect to W* when any connected component of $\mathbb{F}$ contains precisely one element of W.

Example of a spanning

forest w.r.t. $W = \{s_1, s_2, t\}$

Each component

marked by different color



Forest-generated (IRFC and PW) objects

$G = \langle C, E, w \rangle$ – weighted graph, $\emptyset \neq W \subset C$, $S \subset W$

Definition (Forest-generated object)

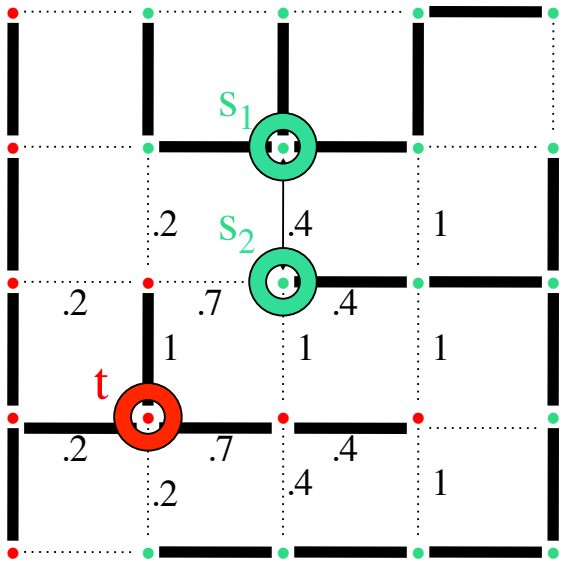
For a spanning forest $\mathbb{F}$ w.r.t. W and $S \subset W$,
$P(S, \mathbb{F})$ is a union of all components of $\mathbb{F}$ intersecting S.
Note that $P(S, \mathbb{F}) \in \mathcal{P}(S, T)$ for $T = W \setminus S$.

Example (green vertices) of

$P(S, \mathbb{F})$ with $S = \{s_1, s_2\}$.

Outputs of the algorithms we will discuss, $\mathrm{GC}_{\mathrm{sum}}$ and PW,

are in the $P(S, \mathbb{F})$ format.



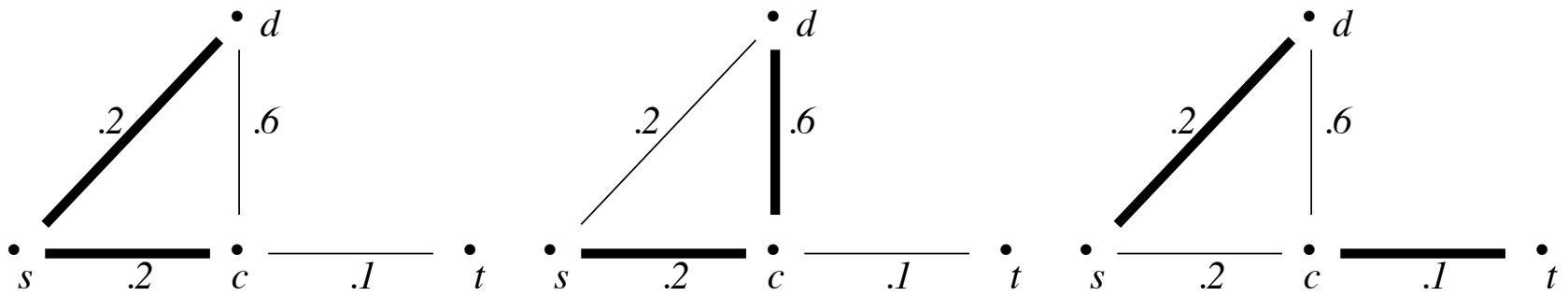
Optimal Path Forest, OPF

Definition (Optimal Path Forest, OPF)

For a path $p = \langle c_1, \ldots, c_k \rangle$ in G let $\mu(p) = \min_{i<k} w(\{c_k, c_{k+1}\})$, the *weakest link* of p.

A forest $\mathbb{F}$ w.r.t. W is *path-optimal* provided for every $c \in C$, the unique path p_c in $\mathbb{F}$ from W to c is μ-optimal in G, i.e., $\mu(p_c) \geq \mu(p)$ for any path p in G from W to c.

For OPF $\mathbb{F}$ w.r.t. W, $\mu(p_c) = \mu^C(c, W)$ for every $c \in C$
(with μ^C in the Fuzzy Connectedness sense)



(g) OPF, $W = \{s, t\}$ (h) another OPF (i) not OPF

GC_{max} algorithm and IRFC

Theorem ([KC *et al.*] OPF object minimizing ε^{max})

There exists the smallest $P_{min} \in \mathcal{P}(S,T)$ in form $P(S,\mathbb{F})$, where $\mathbb{F}$ is an OPF w.r.t. $S \cup T$.

$\mathbb{F}$ is found by GC_{max}, a version of Dijkstra's shortest path algorithm, in a linear time w.r.t. $|C| + M$, where M is the size of the range of w.

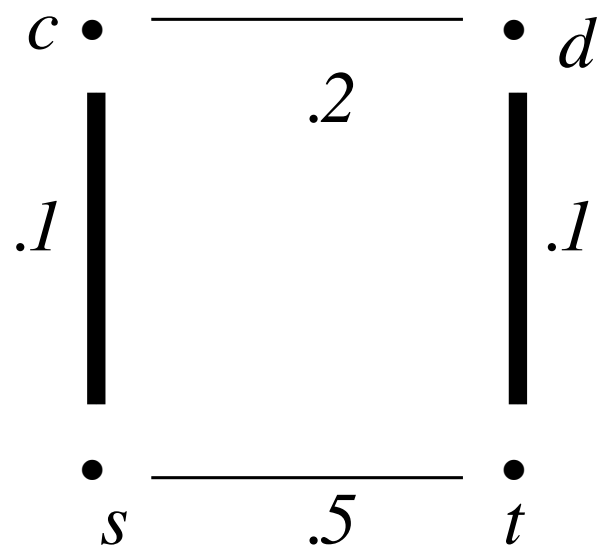
In practice, $O(|C| + M) = O(|C|)$.

The object P_{min}, returned by GC_{max}, coincides with the Iterative Relative Fuzzy Connectedness, IRFC, object.

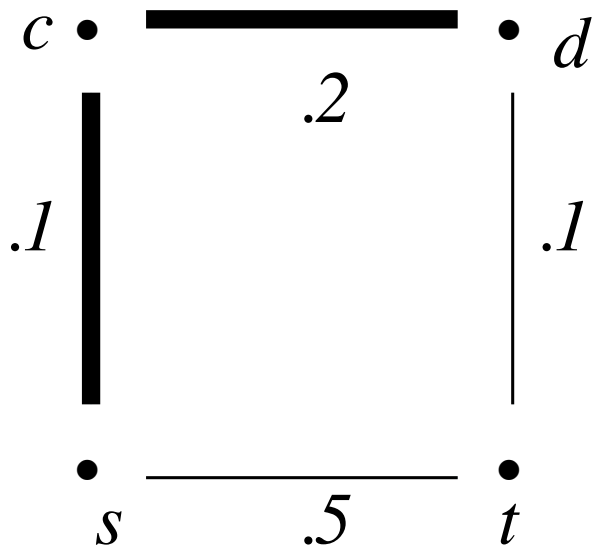
Maximal Spanning Forest, MSF

Definition (Maximal Spanning Forest, MSF)

A forest $\mathbb{F} = \langle C, E' \rangle$ w.r.t. W is *maximal spanning* provided $\sum_{e \in E'} w(e) \geq \sum_{e \in \hat{E}'} w(e)$ for every forest $\hat{\mathbb{F}} = \langle C, \hat{E}' \rangle$ w.r.t. W



(j) OPF w.r.t. $\{s, t\}$, not MSF



(k) MSF and OPF

Theorem ([Audigier & Lotufo], [Cousty *et al.*])

Every MSF is OPF, but not the other way around.

MSF and Power Watershed, PW, algorithm

Theorem ([C. Couprie *et al.*] PW output as MSF)

PW algorithm returns $P(S, \mathbb{F})$ for a MSF $\mathbb{F}$ w.r.t. $S \cup T$.

$\mathbb{F}$ is found by PW via a complicated version of Kruskal's algorithm and, locally, Random Walker algorithm.

Since

- IRFC object is indicated by OPF,
- PW object is indicated by MSF, and
- every MSF is OPF

What is the relation between IRFC and PW objects?

New results on GC_{max}, MSF, and OPF

Theorem ([KC *et al.*] MSF vs OPF)

If P_{min} is the output of GC_{max} (the smallest $P(S, \mathbb{F})$, with with $\mathbb{F}$ is being OPF w.r.t. $S \cup T$), then $P_{min} = P(S, \hat{\mathbb{F}})$ for some MSF $\hat{\mathbb{F}}$.

If $\mathbb{F}$ is a MSF w.r.t. $S \cup T$, then $P(S, \mathbb{F})$ minimizes energy ε^{max} (in $\mathcal{P}(S, T)$).

$P(S, \mathbb{F})$, with $\mathbb{F}$ being OPF w.r.t. $S \cup T$, need not minimize ε^{max}.

In other words

$$P_{min} \in \mathcal{P}_{MSF}(S, T) \subset \mathcal{P}_{OPF}(S, T) \cap \mathcal{P}_{\varepsilon^{max}}(S, T),$$

where $\mathcal{P}_{MSF}(S, T) = \{P(S, \mathbb{F}) : \mathbb{F} \text{ is MSF}\}$, similarly for OPF, and $\mathcal{P}_{\varepsilon^{max}}(S, T)$ is the set of all ε^{max}-optimizers.

Outline of Part 2: Delineating objects via ℓ_p energies

Outline of the proof of Main Theorem

- Describe Dijkstra's algorithm that gives OPF $\mathbb{F}$ with $P_{\min} = \mathcal{P}(S, \mathbb{F})$. Notice, it is the smallest set in $\mathcal{P}_{OPF}(S, T)$.
- Use Kruskal's algorithm to find MSF $\hat{\mathbb{F}}$ with $P_{\min} = \mathcal{P}(S, \hat{\mathbb{F}})$.
- Show that $\mathcal{P}(S, \hat{\mathbb{F}}) \in \mathcal{P}_{\varepsilon^{\max}}(S, T)$ whenever $\hat{\mathbb{F}}$ is MSF.
  An argument is a variant of a proof that Kruskal's algorithm indeed returns MSF.
- Give examples, showing that no inclusion can be reversed.

Dijkstra's algorithm DA: standard vs our version

$G = \langle C, E, w \rangle$, $\mathbb{F}$ generated forest w.r.t. W, $S \subset W \subset C$
p_c – unique path in $\mathbb{F}$ from W to $c \in C$

- Standard DA "grows" tree from a single source set W.
  We use DA to grow forest with a multiple sources set W.
- In standard DA, path p_c has the smallest length.
  (It optimizes path measure "sum of weights of all links.")
  We use DA to optimize p_c w.r.t. "weakest link measure" μ.
- Newest variation:
  We insure that $P_{\min} = P(\mathbb{F}, S)$ is the smallest possible.
  No control of algorithm's output among $\mathcal{P}_{\varepsilon^{\max}}(S, T)$ was insurable before introduction of $\mathrm{GC}_{\max}$ (as far as we know).

GC_{max} (i.e., our DA) data structure

- $\mathbb{F}$ is grown from roots, $W = S \cup T$, via adding edges.
- $\mathbb{F}$ is indicated via path-predecessor map Pr:
  $Pr[W] = \{\emptyset\}$, $Pr(c) =$*predecessor of c in* p_c for $c \notin W$
- $R(c)$ indicates root of c: the initial $w \in W$ belonging to p_c
- We use preorder relation $\prec$ on $\mathbb{R} \times C$:

$$\langle x, c \rangle \prec \langle y, d \rangle \iff x < y \text{ or } (x = y \;\&\; d \in T \;\&\; c \notin T)$$

- Initialize $\mu(c) = 1$, $R(c) = c$, $Pr(c) = \emptyset$ for $c \in W$
- Initialize $\mu(c) = -1$, $R(c) = c$, $Pr(c) = c$ for $c \in C \setminus W$
- Insert every $c \in C$ into queue Q according to priority $\preceq$

The GC_{max} algorithm

begin

1. *while* Q is not empty *do*
2. remove from Q a $\preceq$-maximal spel c;
3. *for* every d with $\{c, d\} \in E$ *do*
4. *if* $\langle \mu(d), R(d) \rangle \prec \langle \min\{\mu(c), w_{\{c,d\}}\}, R(c) \rangle$ *then*
5. set $\mu(d) = \min\{\mu(c), w_{\{c,d\}}\}$;
6. set $R(d) = R(c)$ and $Pr(d) = c$;
7. remove temporarily d from Q;
8. push d to Q with the current values of μ and R;
9. *endif*;
10. *endfor*;
11. *endwhile*;
12. return $\mu(\cdot, W) = \mu(\cdot)$, $\mathbb{F}$ indicated by Pr, $P_{\min} = P(S, \mathbb{F})$;

end

Properties of GC_{max}; correctness

line 2: Each $c \in C$ is removed precisely once from Q

- with $\mu(c) = \mu(c, W)$
- with $\prec$-maximal value of $\langle \mu(c), R(c) \rangle$

Proof: If the above fails for a $c \in C$ and c comes from the fist execution of line 2 when this happens, then, in earlier execution of lines 4-9, the value $\langle \mu(c), R(c) \rangle$ would have been increased.

So, indeed $\mathbb{F}$ is OPF and
$P_{\min} = \mathcal{P}(S, \mathbb{F})$ is the $\subset$-smallest element of $\mathcal{P}_{OPF}(S, T)$.

Next we show that $P_{\min} = P(S, \hat{\mathbb{F}})$ for some MSF $\hat{\mathbb{F}}$

Kruskal's algorithm KA

Kruskal's algorithm creates MSF $\hat{\mathbb{F}}$ for $G = \langle C, E, w \rangle$ as follows:

- it lists all edges of the graph in a queue Q such that their weights form a non-increasing sequence;
- it removes consecutively the edges from Q, adding to $\hat{\mathbb{F}}$ those, which addition creates, in the expanded $\hat{\mathbb{F}}$, neither a cycle nor a path between different vertices from W; other edges are discarded.

This schema has a leeway in choosing the order of edges in Q: those that have the same weight can be ordered arbitrarily.

This leeway will be exploited in the next proof.

Construction of MSF $\hat{\mathbb{F}}$ with $P_{\min} = P(S, \hat{\mathbb{F}})$

Put $B = \mathrm{bd}(P(S, \mathbb{F}))$.

Insert every $e \in E$ into queue Q such that:

- the weights of $e \in Q$ are in a non-increasing order;
- among the edges with the same weight, all those from $E \setminus B$ precede all those from B.

Apply Kruskal's algorithm to this Q to get MSF $\hat{\mathbb{F}}$.

$\hat{\mathbb{F}}$ is an MSF by the power of Kruskal's algorithm.

To prove that $P(S, \hat{\mathbb{F}}) = P(S, \mathbb{F})$ it is enough to show that $\hat{\mathbb{F}} \cap B = \emptyset$.

$\hat{\mathbb{F}}$ is disjoint with $B = \mathrm{bd}(P(S, \mathbb{F}))$

Let $e = \{c, d\} \in B = \mathrm{bd}(P(S, \mathbb{F}))$, $c \in P(T, \mathbb{F})$. We show that:

In KA, adding e to $\hat{\mathbb{F}}$ would create a path from S to T.

Let p_c and p_d be the paths in $\mathbb{F}$ from W to c and d. Then

$$\mu(p_c) \geq w_e \text{ and } \mu(p_d) \geq w_e. \quad (1)$$

Proof: If $\mu(p_c) > \mu(p_d)$, then $w_e \leq \mu(p_d)$, since otherwise $\mu(p_d) < \min\{\mu(p_c), w_e\} \leq \mu(d, W)$, contradicting optimality of p_d.

Similarly, $\mu(p_c) < \mu(p_d)$ implies $w_e \leq \mu(p_c)$.

Finally, $\mu(p_c) = \mu(p_d)$ implies $w_e < \mu(p_c) = \mu(p_d)$, since otherwise $\mathrm{GC}_{\max}$ (during the execution of lines 6-8 for c and d) would reassign d to $P(T, \mathbb{F})$, contradicting $d \in P(S, \mathbb{F})$.
So, (1) is proved.

$\hat{\mathbb{F}}$ is disjoint with $B = \mathrm{bd}(P(S, \mathbb{F}))$, continuation

For $e = \{c, d\} \in B = \mathrm{bd}(P(S, \mathbb{F}))$, $c \in V \setminus P(S, \mathbb{F})$, we show:

In KA, adding e to $\hat{\mathbb{F}}$ would create a path from S to T.

For paths p_c and p_d in $\mathbb{F}$ from W to c and d,

$$\mu(p_c) \geq w_e \text{ and } \mu(p_d) \geq w_e.$$

Let $E' = \{e' \in E : w_{e'} \geq w_e\} \setminus B$. Then, $\hat{\mathbb{F}} \cap E'$ is already constructed by KA. It is enough to show that

In $\hat{G} = \langle V, \hat{\mathbb{F}} \cap E' \rangle$ there is path $\hat{p}_d$ from S to d and $\hat{p}_c$ from T to c.

Proof. The component $\mathbb{C}$ of d in $\hat{G}$ intersects S, as otherwise there is an $\hat{e} \in p_d \subset E'$ only one vertex of which intersects $\mathbb{C}$ and $\hat{e} \in E'$ would have been added to $\hat{\mathbb{F}}$, but was not. So, indeed, there is $\hat{p}_d$ as claimed. Similarly, for $\hat{p}_c$. QED

If $\mathbb{F}$ is an MSF, then $P(S, \mathbb{F})$ minimizes $\varepsilon^{\max}$

Let $\mathbb{F}$ be an MSF and $P = P(S, \mathbb{F})$. Note that

$$\varepsilon_{\min} \stackrel{\text{def}}{=} \{\varepsilon^{\max}(P) \colon P \in \mathcal{P}(S, T)\} = \max\{\mu(p) \colon p \text{ is from } S \text{ to } T\}$$

We need to show that $\varepsilon^{\max}(P) \leq \varepsilon_{\min}$. Assume it is not.

Then, there is an $e = \{c, d\} \in E$ with $c \in P = P(S, \mathbb{F}) \cap \mathrm{bd}(P)$ for which $w_e > \varepsilon_{\min}$. Let p_c and p_d be the paths in $\mathbb{F}$ from W to c and d. Then either $\mu(p_c) < w_e$ or $\mu(p_d) < w_e$; otherwise there is path p from S to T with $\mu(p) = w_e > \varepsilon_{\min}$, a contradiction.

Assume that $\mu(p_c) < w_e$. Then $p_c = \langle c_1, \ldots, c_k \rangle$ with $k > 1$ and $e' = \{c_{k-1}, c_k\}$ has weight $\leq \mu(p_c) < w_e$. But then $\mathbb{F}' = \mathbb{F} \cup \{e\} \setminus \{e'\}$ is a spanning forest w.r.t. W with $w(\mathbb{F}') = w(\mathbb{F}) + w_e - w_{e'} > w(\mathbb{F})$, contradicting that $\mathbb{F}$ is MSF. QED

Summary

We proved that $\text{GC}_{\max}$ algorithm returns OPF $\mathbb{F}$ for which $P(S, \mathbb{F})$ minimizes $\varepsilon^{\max}(P) \stackrel{\text{def}}{=} \max_{e \in \text{bd}(P)} w(e)$ in $\mathcal{P}(S, T)$.

Moreover,

$$P_{\min} \in \mathcal{P}_{MSF}(S, T) \subset \mathcal{P}_{OPF}(S, T) \cap \mathcal{P}_{\varepsilon^{\max}}(S, T),$$

where $\mathcal{P}_{MSF}(S, T) = \{P(S, \mathbb{F}) \colon \mathbb{F} \text{ is MSF}\}$, similarly for OPF, and $\mathcal{P}_{\varepsilon^{\max}}(S, T)$ is the set of all $\varepsilon^{\max}$-optimizers.

None of the inclusions can be reversed.

Thank you for your attention!