Optimization: Then and Now

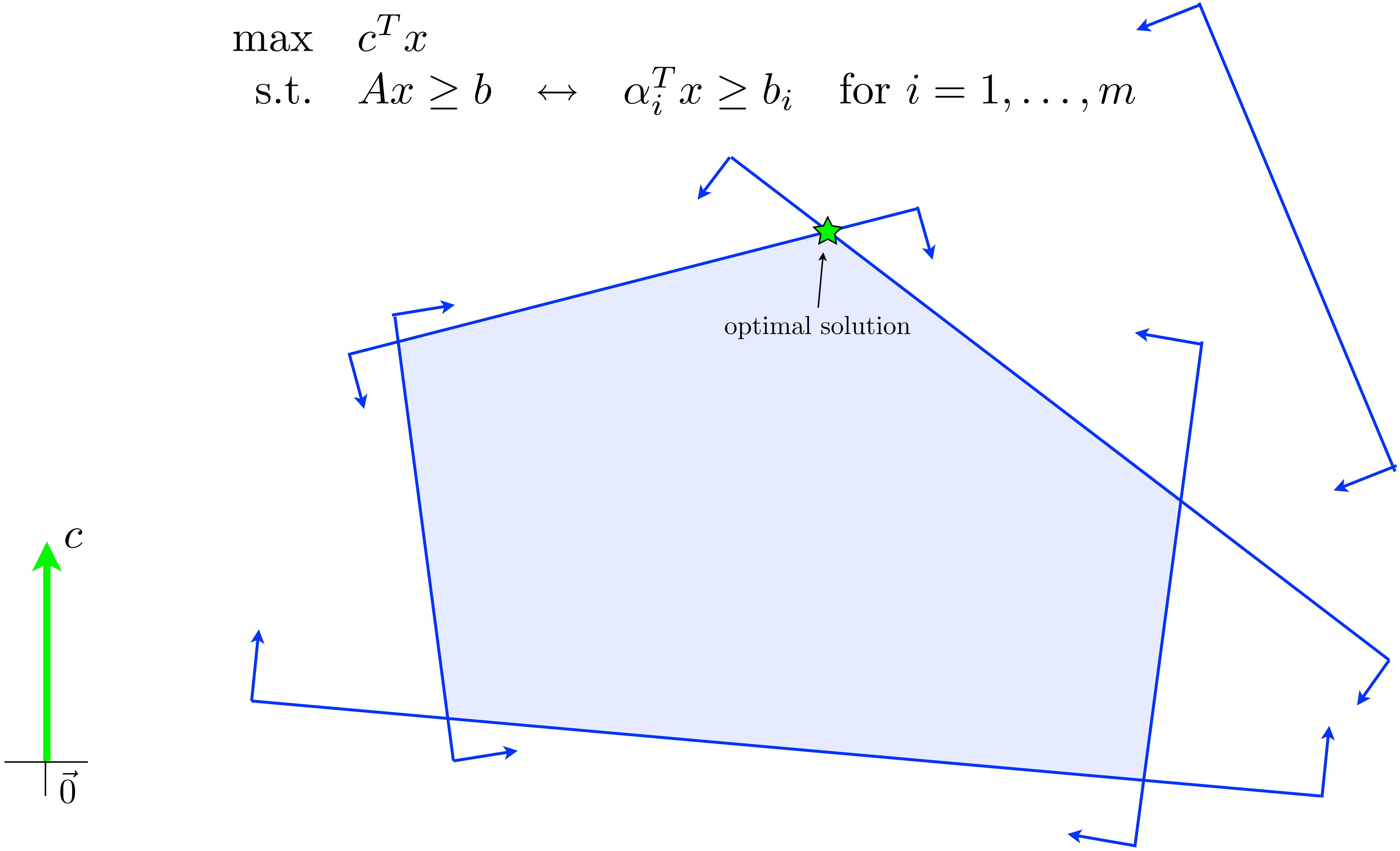
Optimization: Then and Now

~~Optimization: Then and Now~~

Why would a dynamicist be interested in linear programming?

Linear Programming (LP)

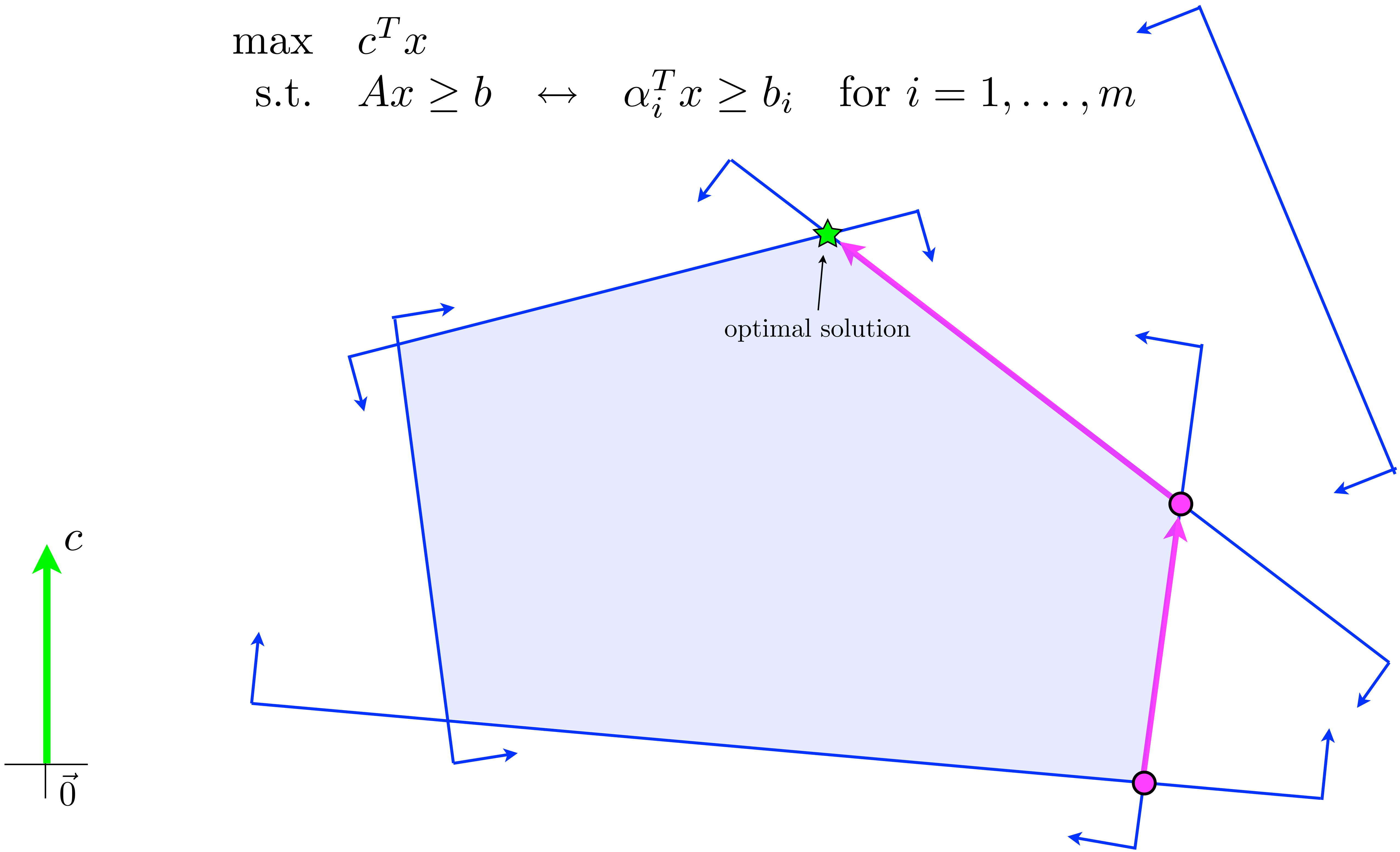
$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & Ax \geq b \quad \leftrightarrow \quad \alpha_i^T x \geq b_i \quad \text{for } i = 1, \ldots, m \end{aligned}$$



Linear Programming (LP)

First general algorithm: The Simplex Method, by George Dantzig (1947)

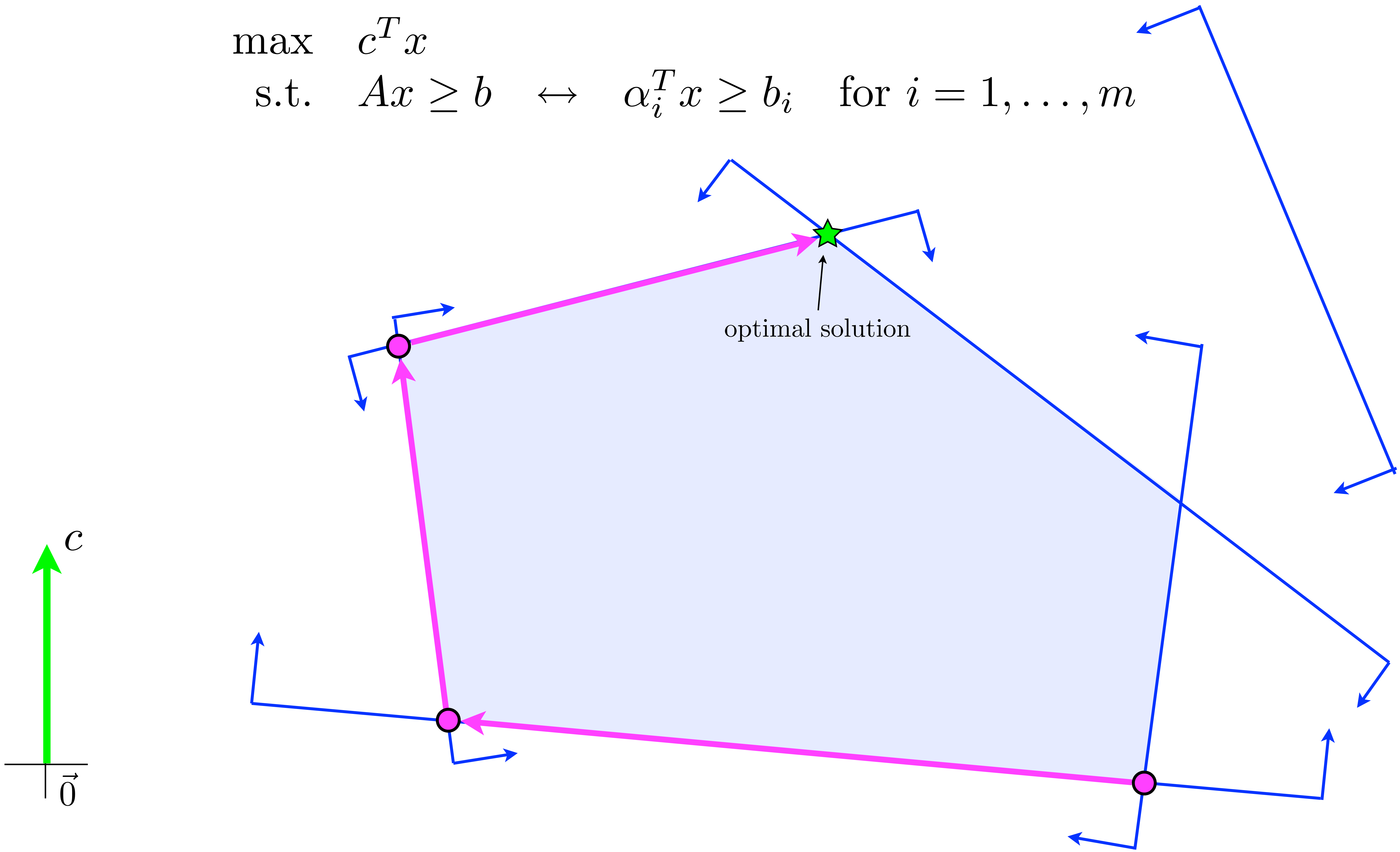
$$\begin{array}{ll} \max & c^T x \\ \text{s.t.} & Ax \geq b \quad \leftrightarrow \quad \alpha_i^T x \geq b_i \quad \text{for } i = 1, \ldots, m \end{array}$$



Linear Programming (LP)

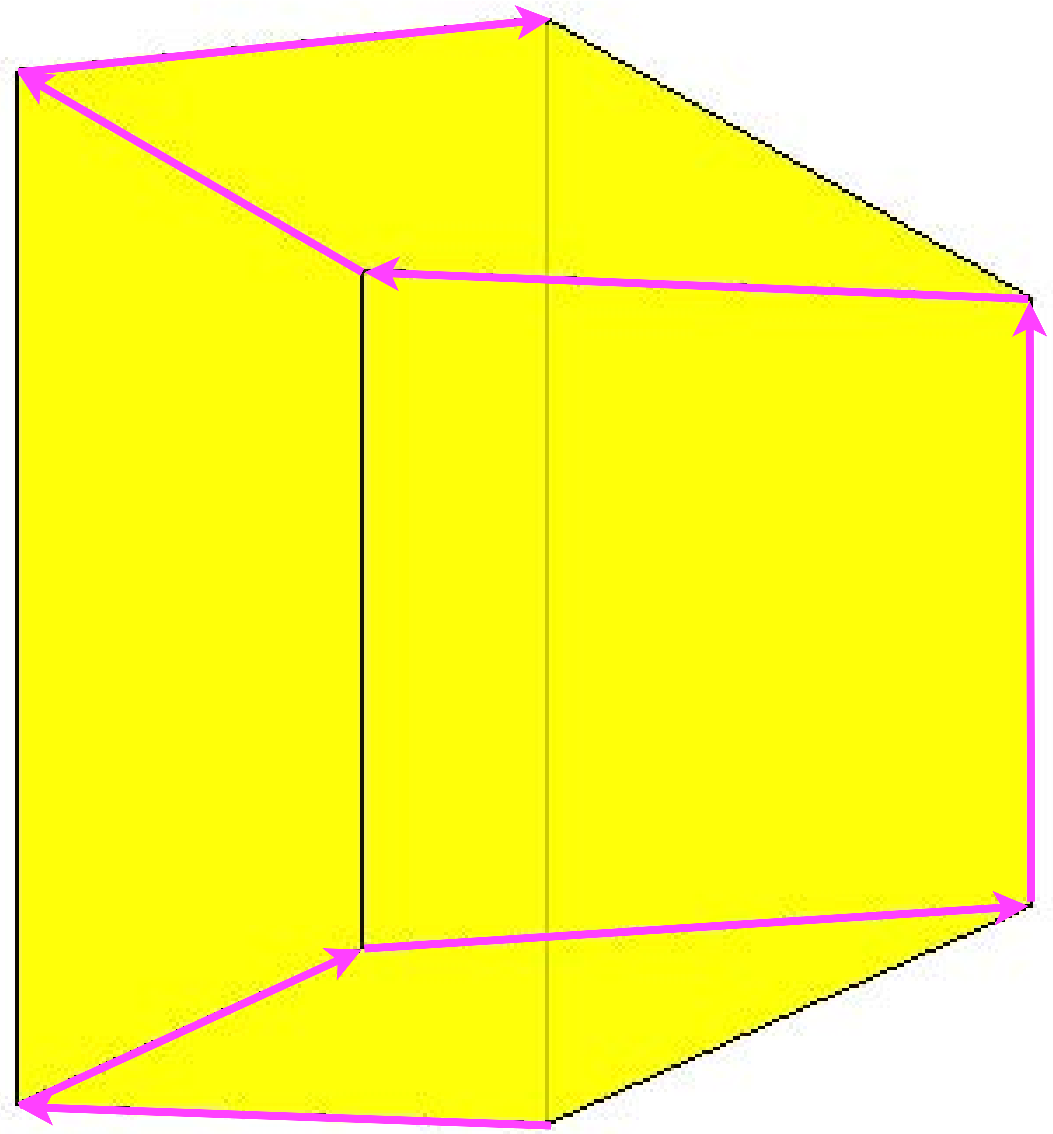
First general algorithm: The Simplex Method, by George Dantzig (1947)

$$\begin{array}{ll} \max & c^T x \\ \text{s.t.} & Ax \geq b \quad \leftrightarrow \quad \alpha_i^T x \geq b_i \quad \text{for } i = 1, \ldots, m \end{array}$$



The "Klee-Minty Cube"

$$\begin{array}{rl} \max & x_n \\ \text{s.t.} & \epsilon \le x_1 \le 1-\epsilon \\ & \epsilon x_j \le x_{j+1} \le 1-\epsilon x_j \quad \text{for } j=1,\ldots,n-1 \end{array}$$



Simplex Method visits all 2^n vertices

(depending on the particular "pivot" rule)

Polynomial-time* algorithms

- The Ellipsoid Method (Khachiyan, 1979)

- "Karmarkar's Algorithm" (Karmarkar, 1984)

  – formally known as the "projective rescaling algorithm"

- Barrier Method

  This algorithm existed long before the others,
  but was proven to be polynomial-time only later, by Gonzaga in 1986

  – who was motivated by work that had established relations
  between the Barrier Method and Karmarkar's Algorithm
  (Gill, Murray, Saunders, Tomlin and Wright (1985))

- Potential Reduction Methods

⋮ Many other algorithms, too.

---

* – polynomial-time, that is, in the Turing model of computation,
not in the Blum-Shub-Smale model!

Whether there exists a BSS polynomial-time LP algorithm remains a major open question.

Interior-Point Methods (IPM's)

- "Karmarkar's Algorithm" (Karmarkar, 1984)

  – formally known as the "projective rescaling algorithm"

  - Barrier Method

    This algorithm existed long before the others,
    but was proven to be polynomial-time only later, by Gonzaga in 1986

    – who was motivated by work that had established relations
    between the Barrier Method and Karmarkar's Algorithm
    (Gill, Murray, Saunders, Tomlin and Wright (1985))

  - Potential Reduction Methods

  ⋮ Many other algorithms, too.

"All IPM's follow the '<u>central path</u>' "

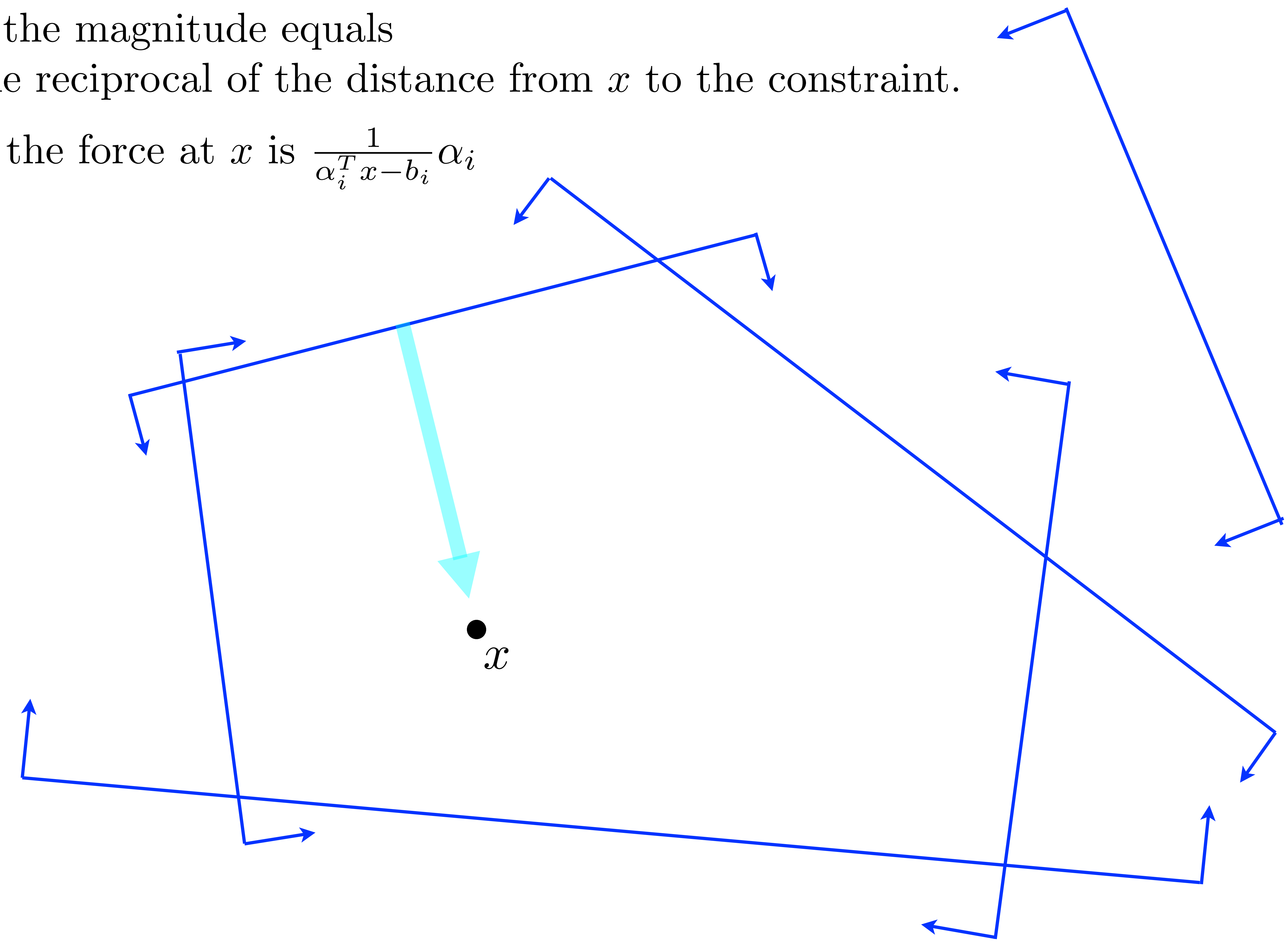
– a.k.a., the "path of analytic centers"

$Ax \geq b \quad \leftrightarrow \quad \alpha_i^T x \geq b_i \quad \text{for } i = 1, \ldots, m$

Think of each constraint as emitting a force that acts on feasible points x.

The direction of the force is perpendicular to the constraint
and the magnitude equals
the reciprocal of the distance from x to the constraint.

I.e., the force at x is $\frac{1}{\alpha_i^T x - b_i} \alpha_i$

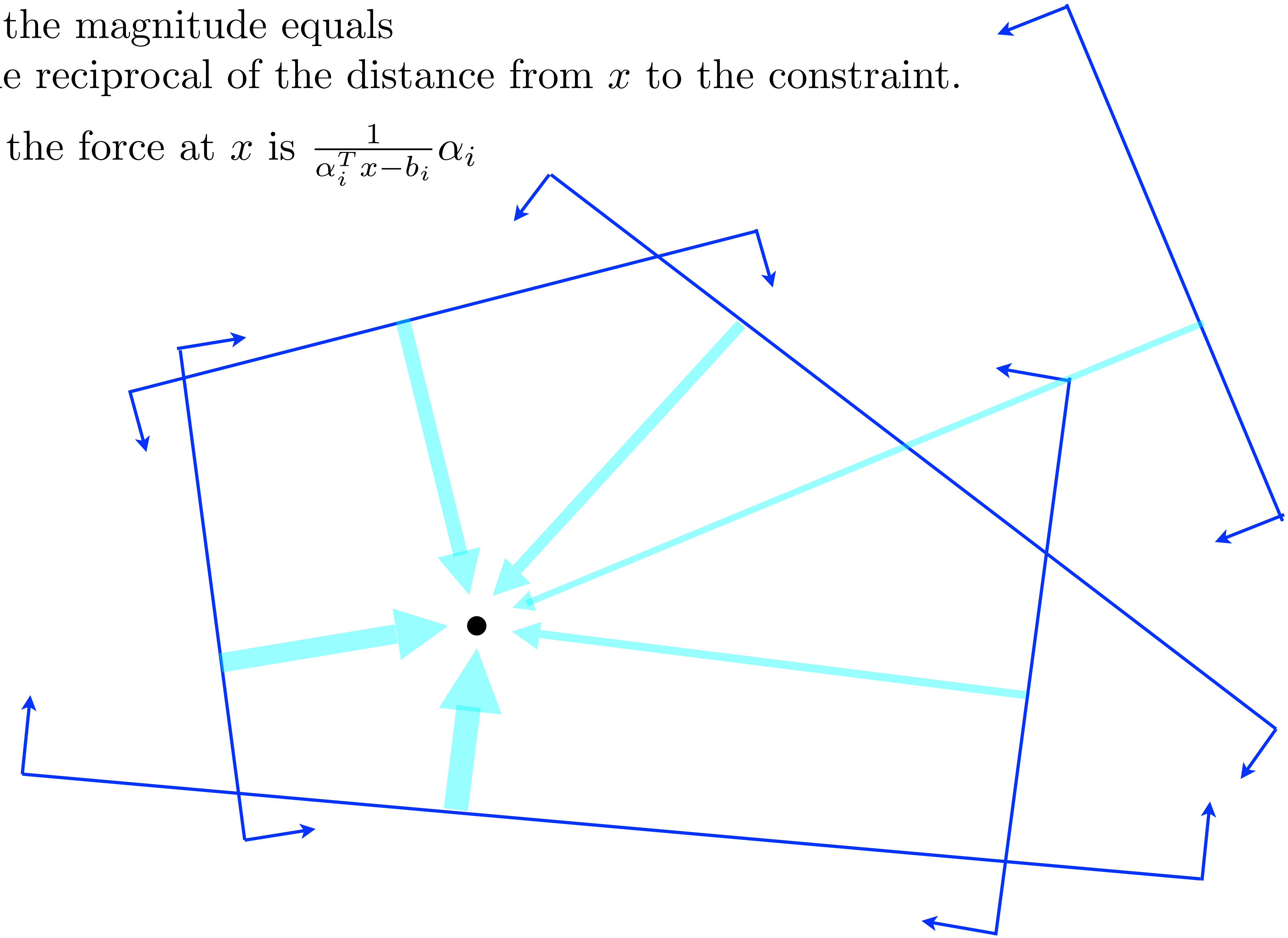


$$Ax \geq b \quad \leftrightarrow \quad \alpha_i^T x \geq b_i \quad \text{for } i = 1, \ldots, m$$

Think of each constraint as emitting a force that acts on feasible points x.

The direction of the force is perpendicular to the constraint
and the magnitude equals
the reciprocal of the distance from x to the constraint.

I.e., the force at x is $\frac{1}{\alpha_i^T x - b_i} \alpha_i$

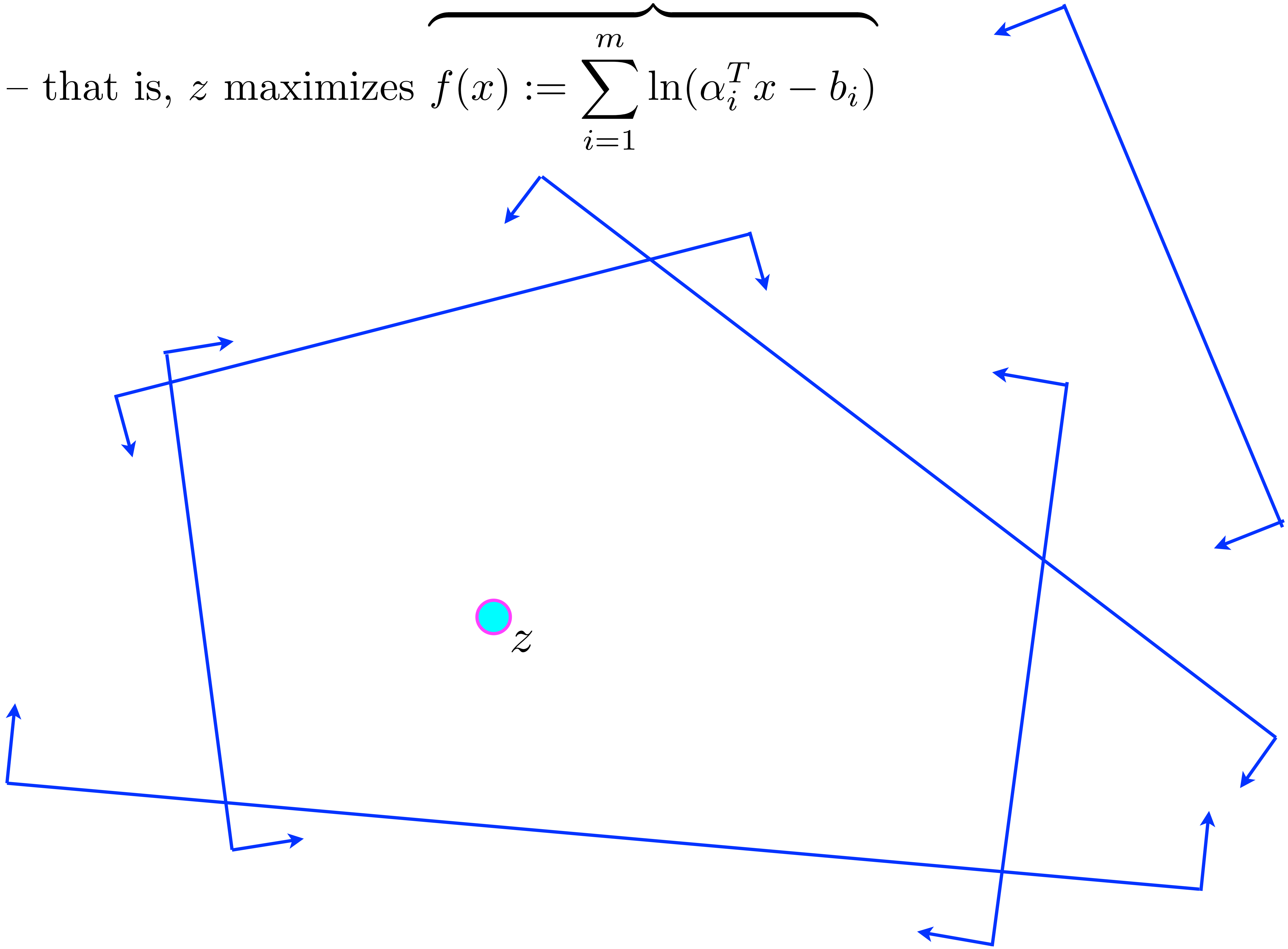


$Ax \geq b \quad \leftrightarrow \quad \alpha_i^T x \geq b_i \quad$ for $i = 1, \ldots, m$

The equilibrium point z is called "the analytic center"

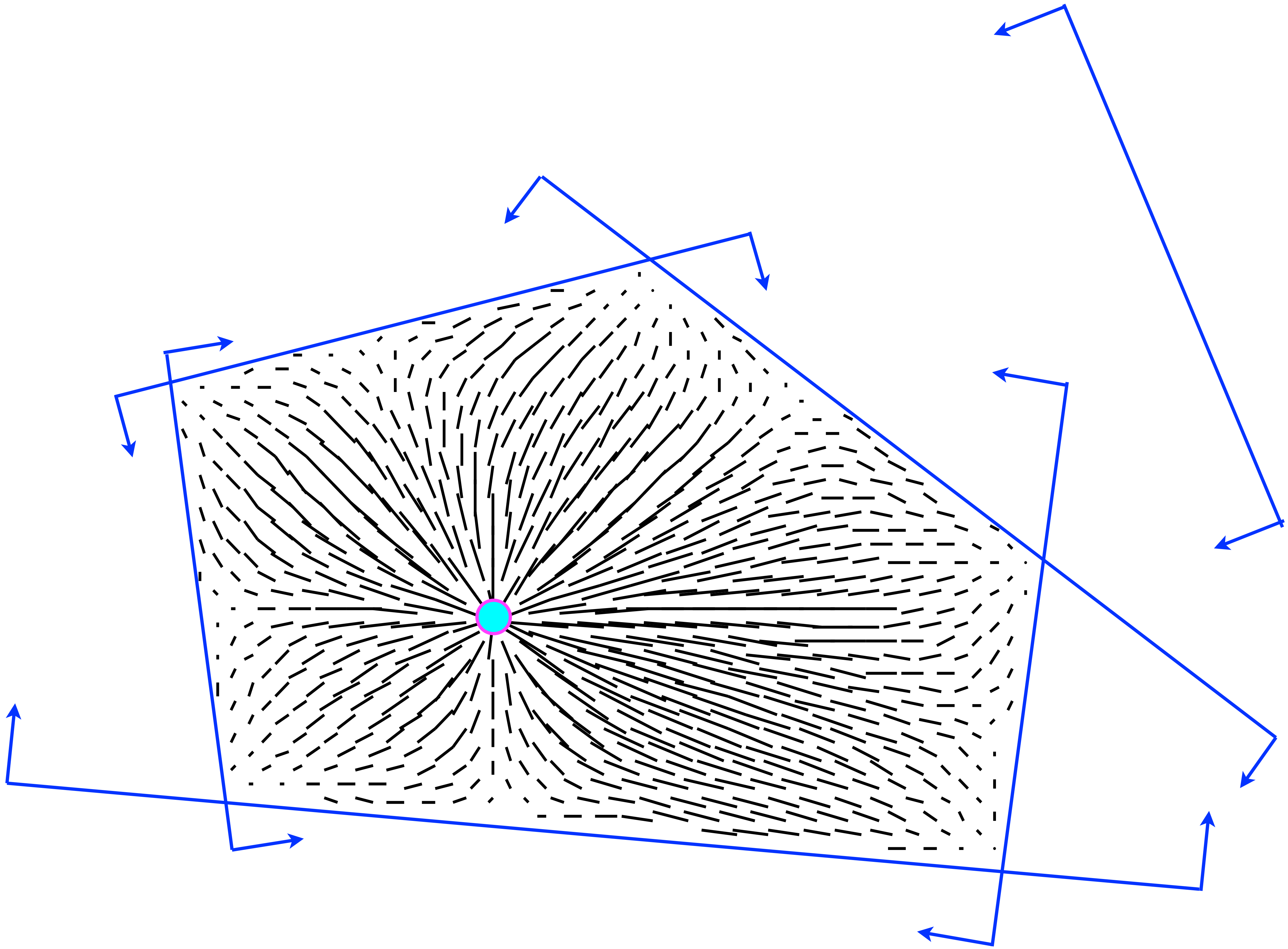
– thus, $\quad 0 = \sum_{i=1}^m \frac{1}{\alpha_i^T z - b_i} \alpha_i$

– that is, z maximizes $f(x) := \overbrace{\sum_{i=1}^{m} \ln(\alpha_i^T x - b_i)}^{\text{a very nice, (strictly) concave function}}$



Analytic center z maximizes $f(x) := \sum \ln(\alpha_i^T x - b_i)$. "barrier function"

Newton's method for maximizing f: $x \mapsto x - (\nabla^2 f(x)^{-1}) \nabla f(x)$

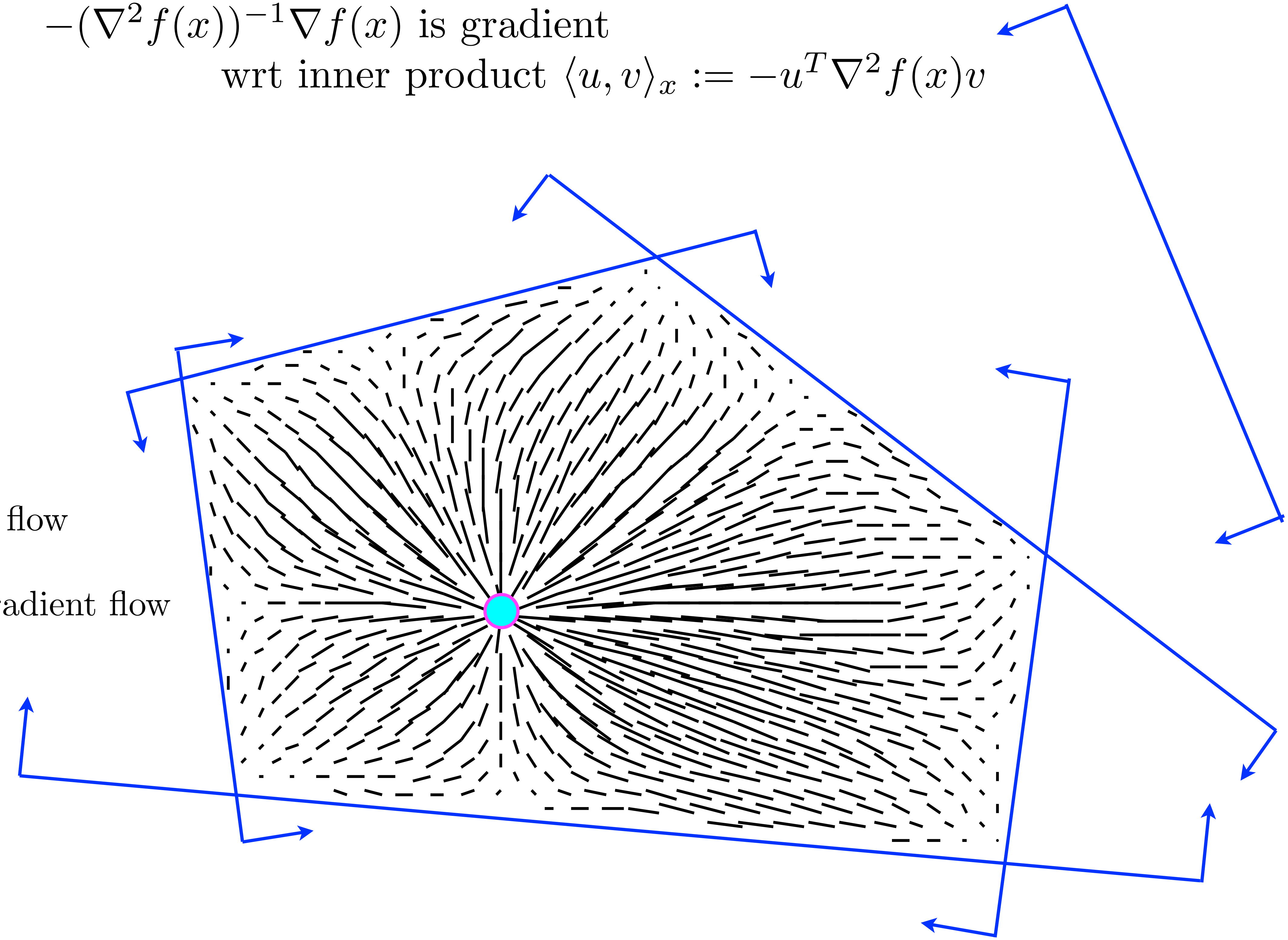


Analytic center z maximizes $f(x) := \sum \ln(\alpha_i^T x - b_i)$. "barrier function"

Newton's method for maximizing f: $x \mapsto x - (\nabla^2 f(x)^{-1})\nabla f(x)$

Note: $-(\nabla^2 f(x))^{-1}\nabla f(x)$ is gradient wrt inner product $\langle u, v\rangle_x := -u^T \nabla^2 f(x) v$

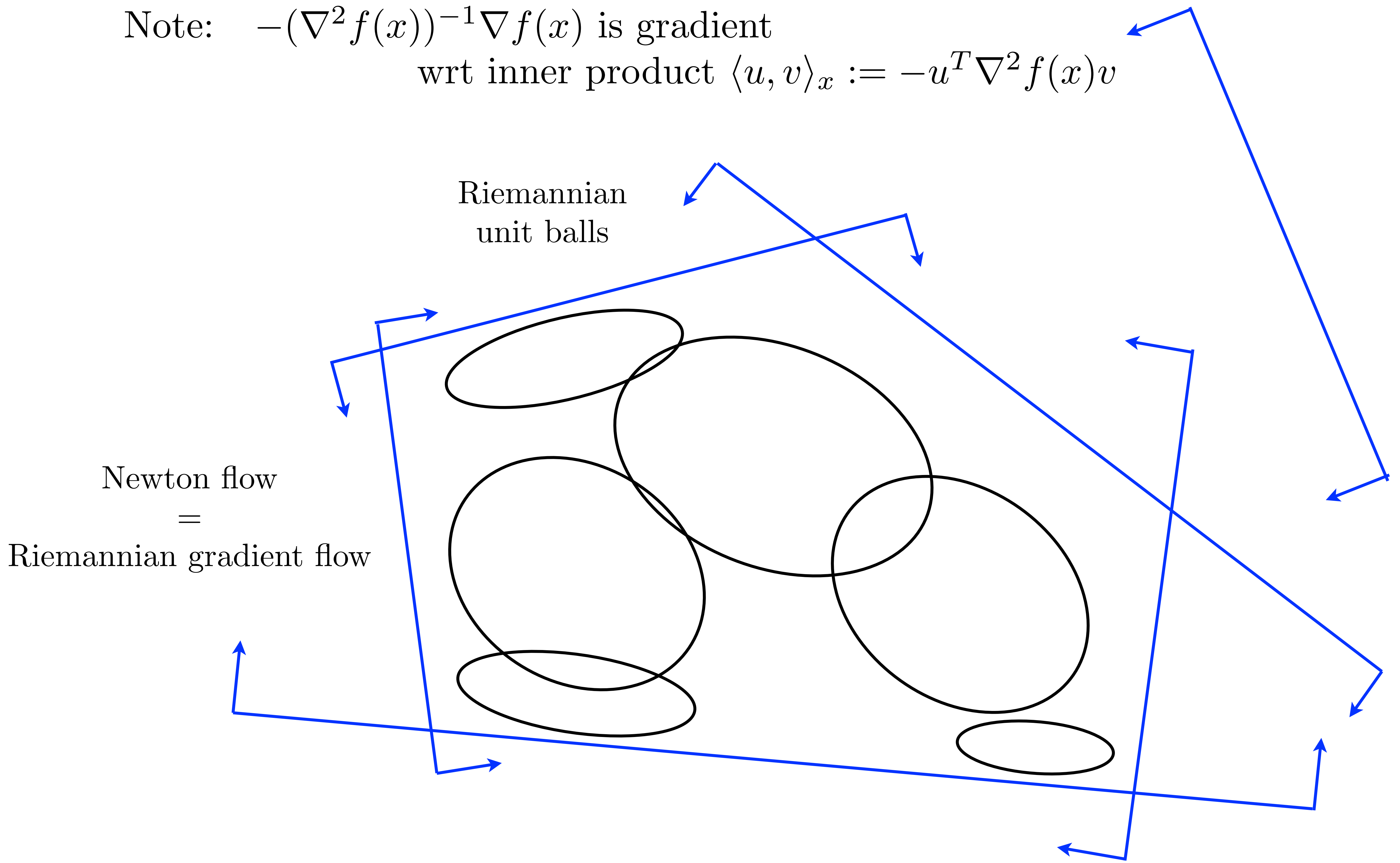
Newton flow = Riemannian gradient flow

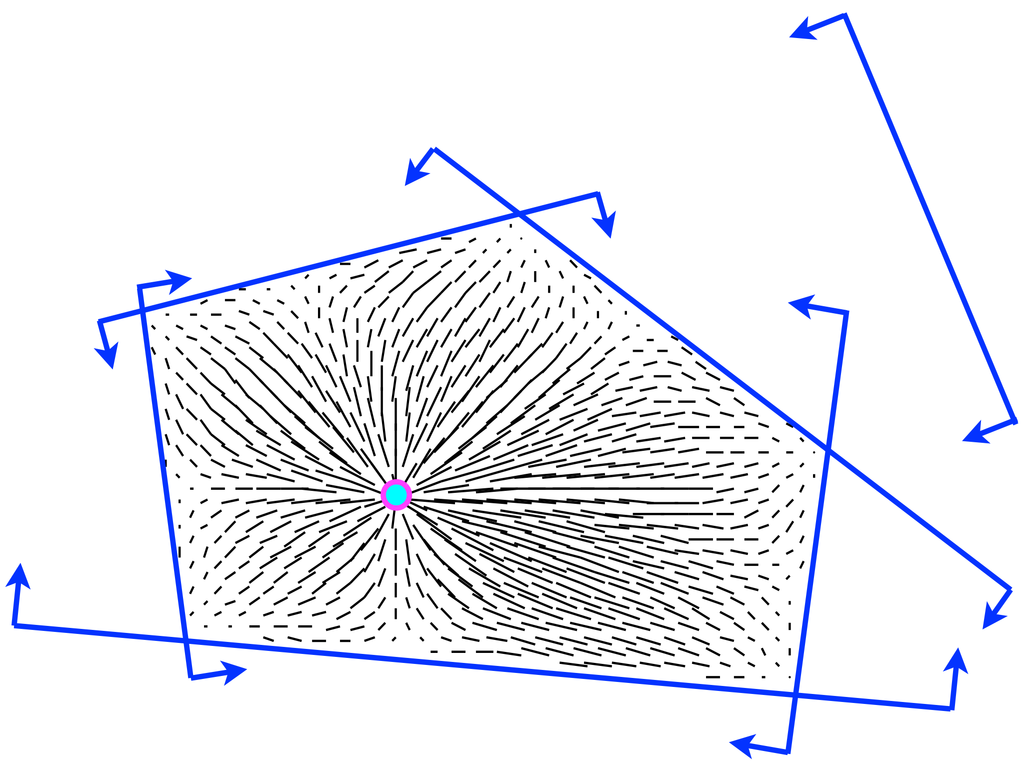


Analytic center z maximizes $f(x) := \sum \ln(\alpha_i^T x - b_i)$. "barrier function"

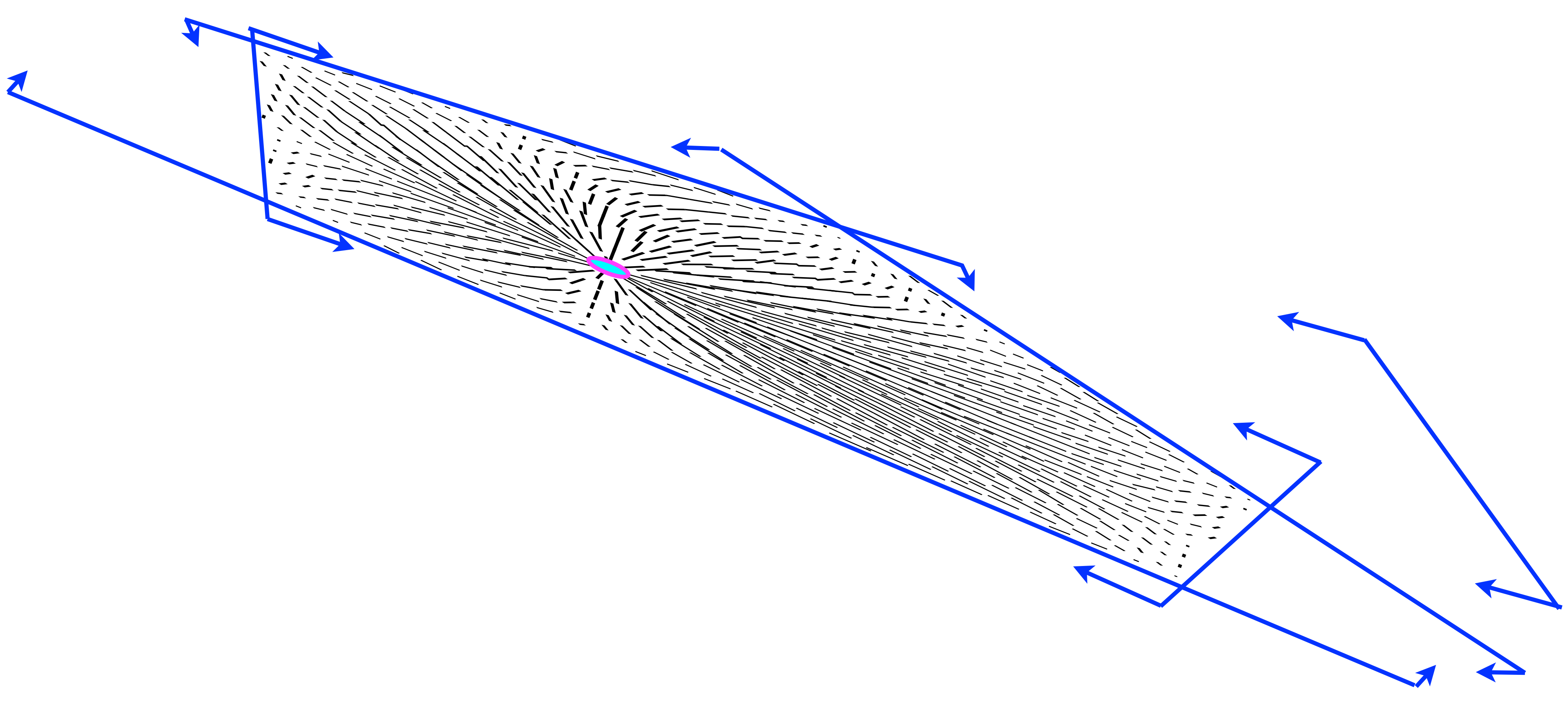
Newton's method for maximizing f: $x \mapsto x - (\nabla^2 f(x)^{-1})\nabla f(x)$

Note: $-(\nabla^2 f(x))^{-1}\nabla f(x)$ is gradient wrt inner product $\langle u, v\rangle_x := -u^T \nabla^2 f(x) v$

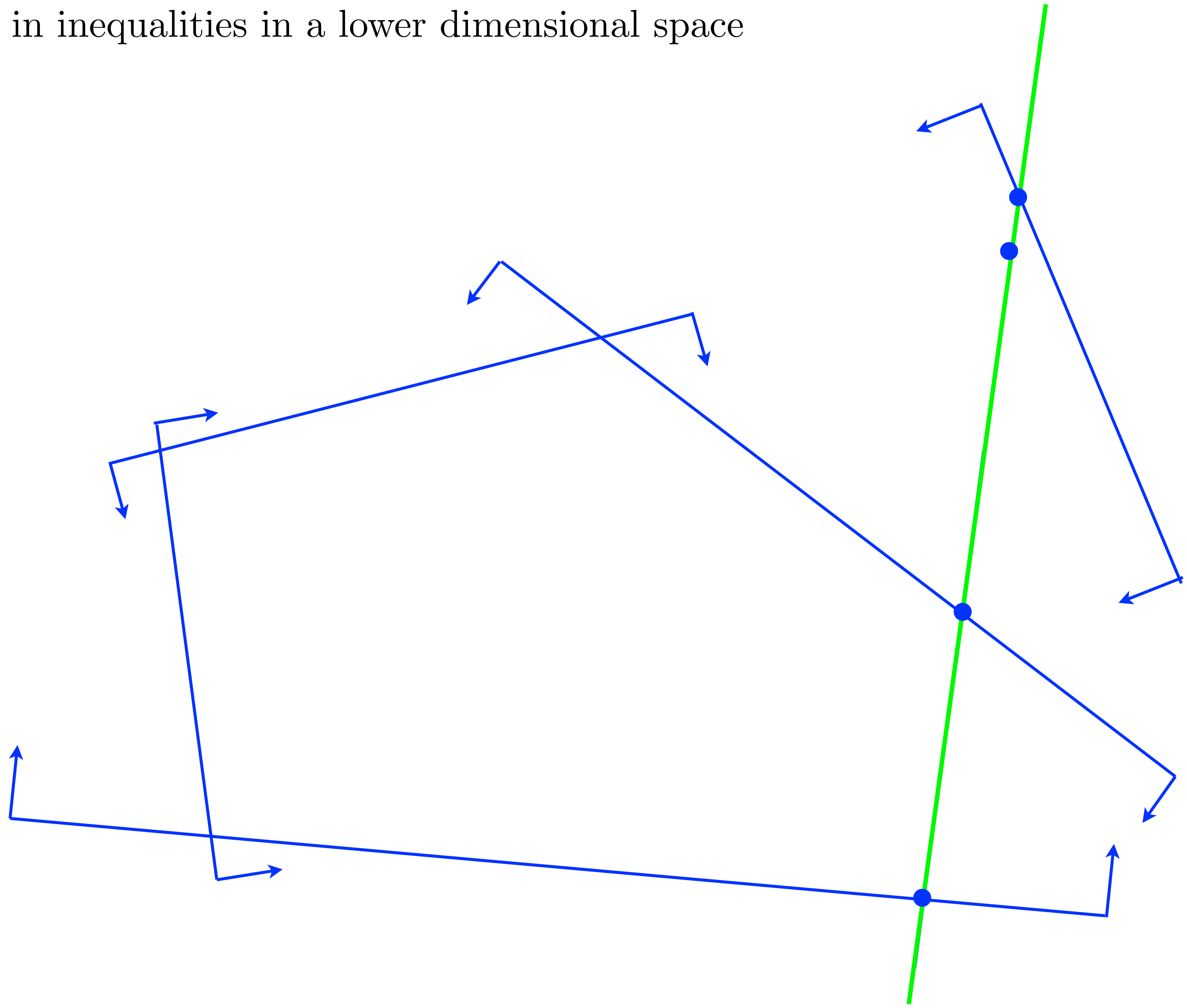




the Newton flow is preserved under invertible affine transformations

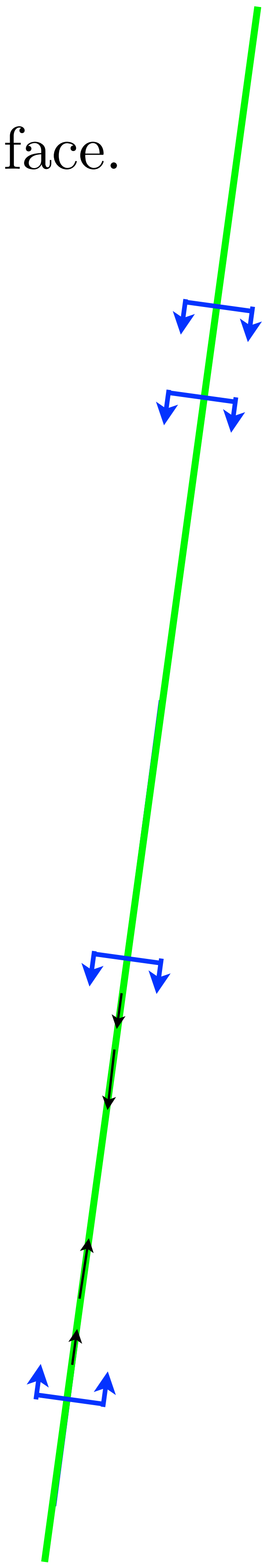


Of course restricting the inequalities $Ax \geq b$ to a supporting hyperplane results in inequalities in a lower dimensional space

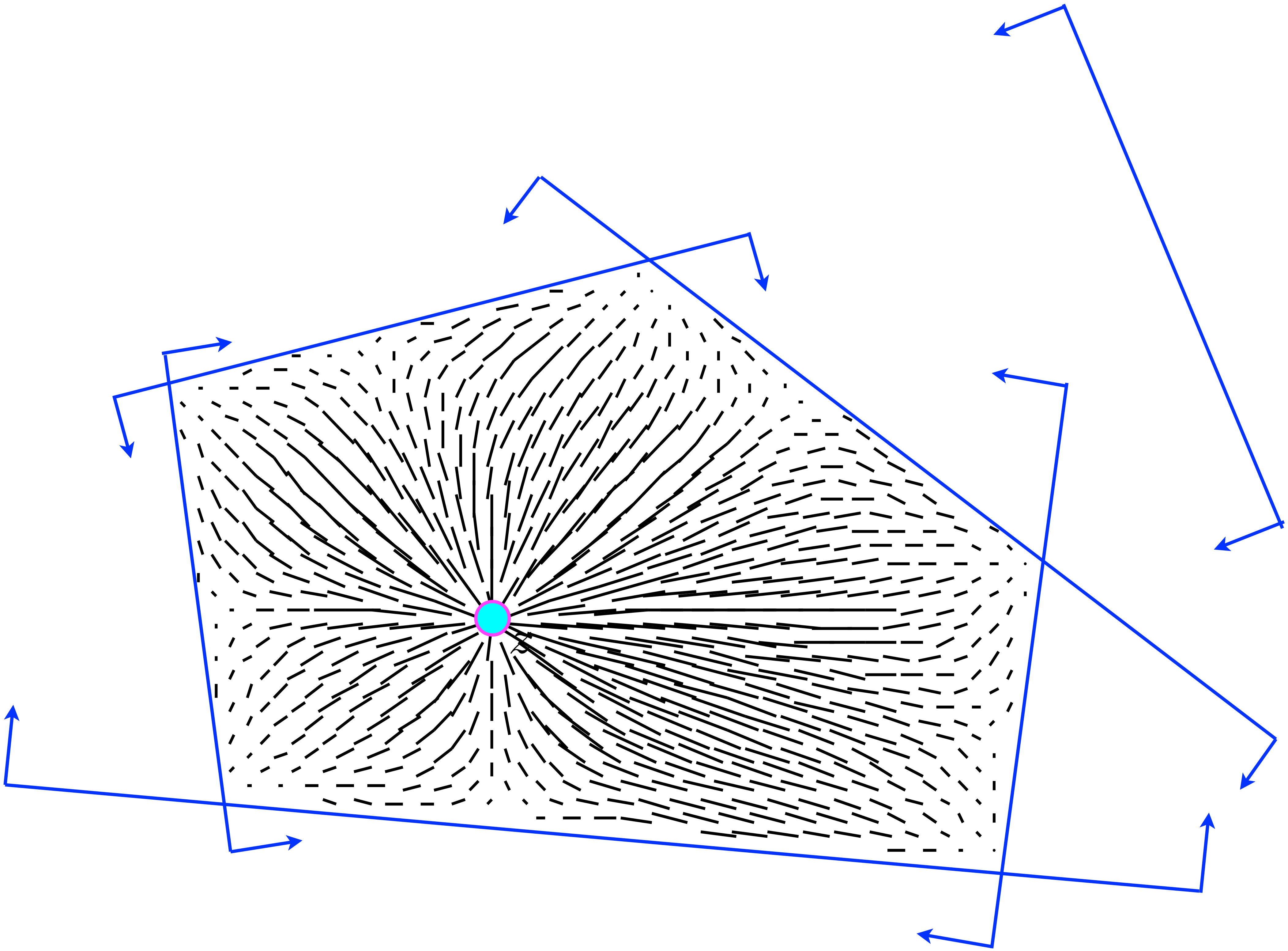


Of course restricting the inequalities $Ax \geq b$ to a supporting hyperplane results in inequalities in a lower dimensional space

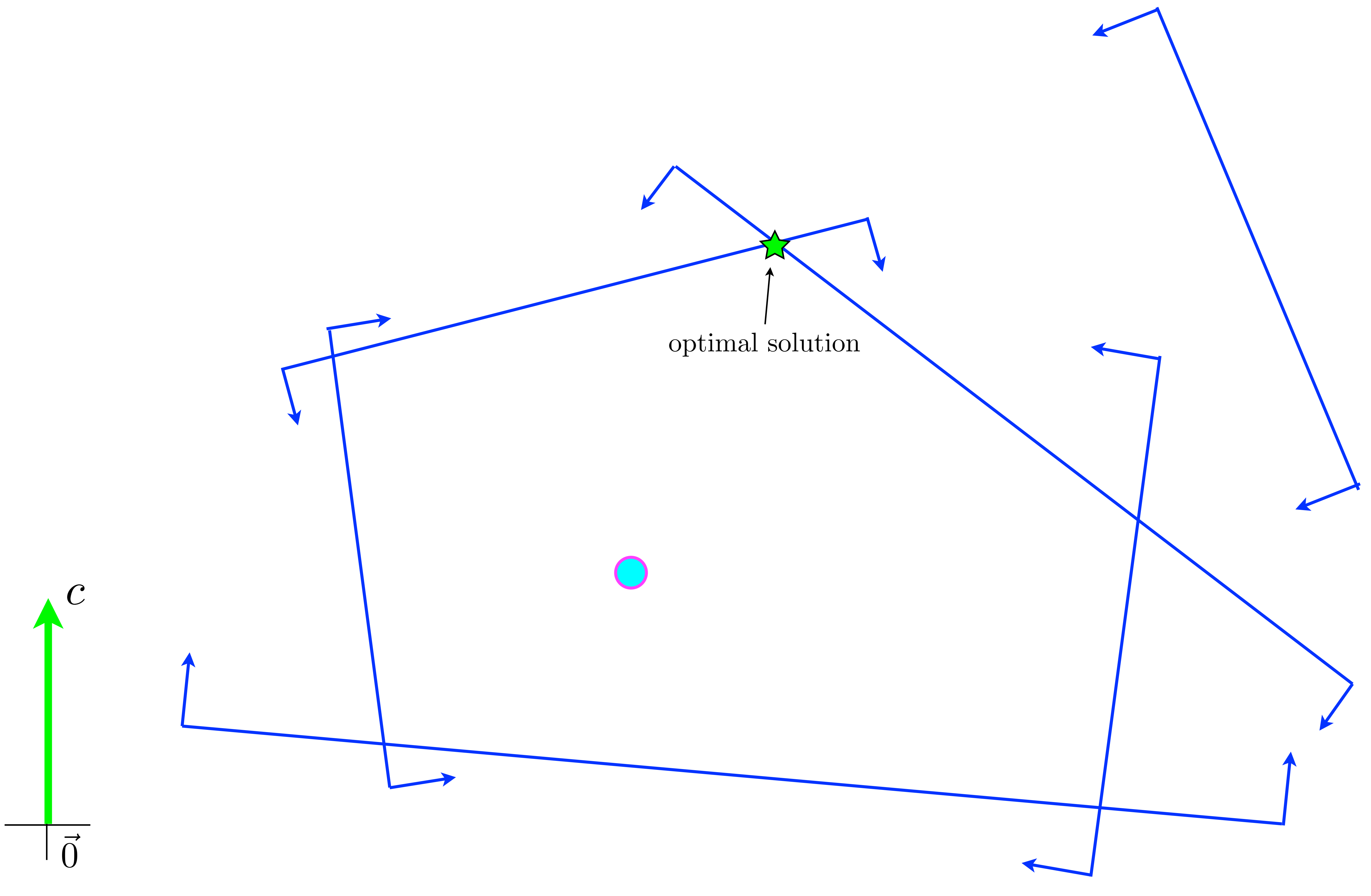
... and hence naturally induces a Newton flow on the face.

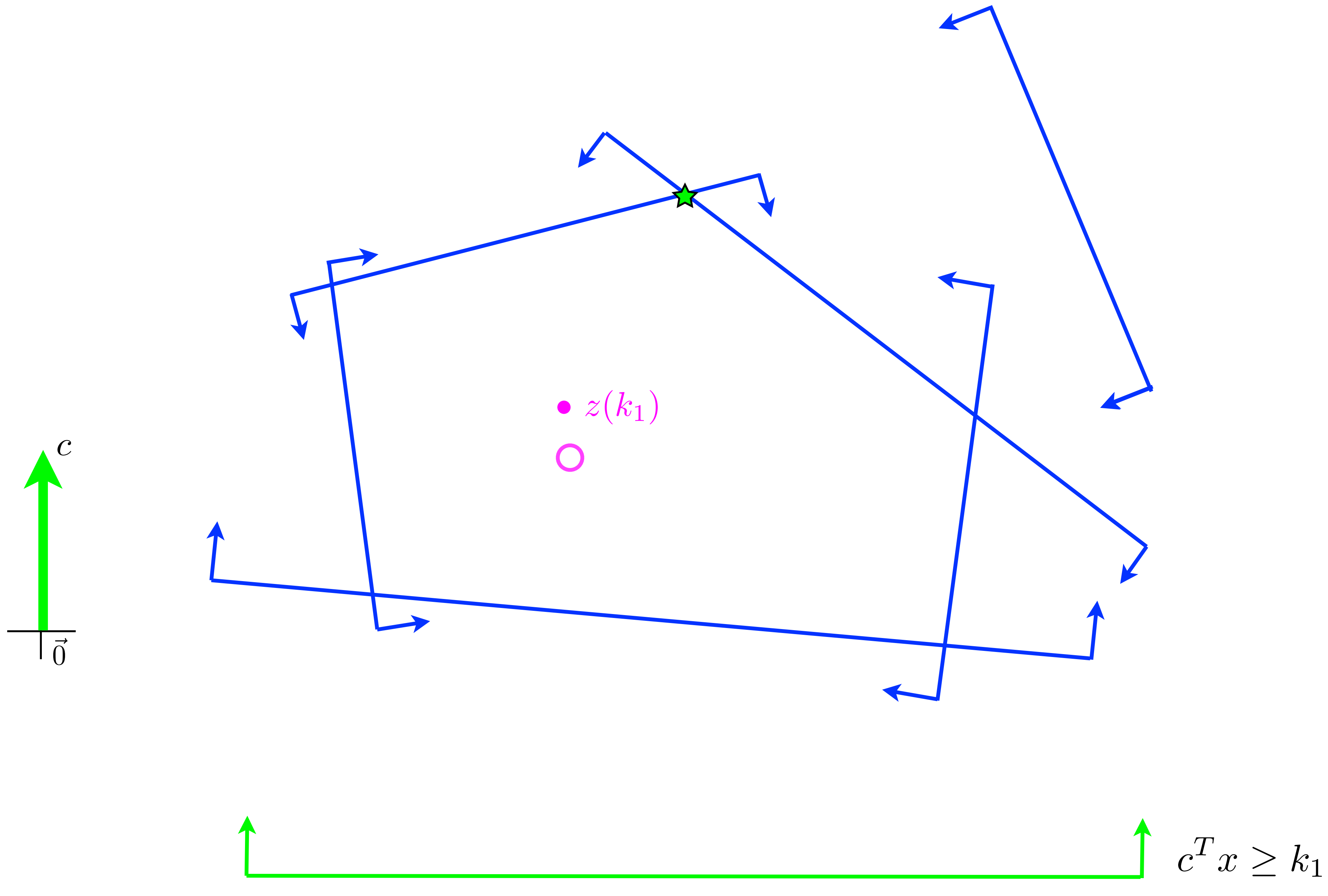


Mike and Jean-Pierre (2004):
For generic (A, b), the Newton flows on the faces analytically extend the Newton flow on the interior.



$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & Ax \geq b \end{aligned}$$

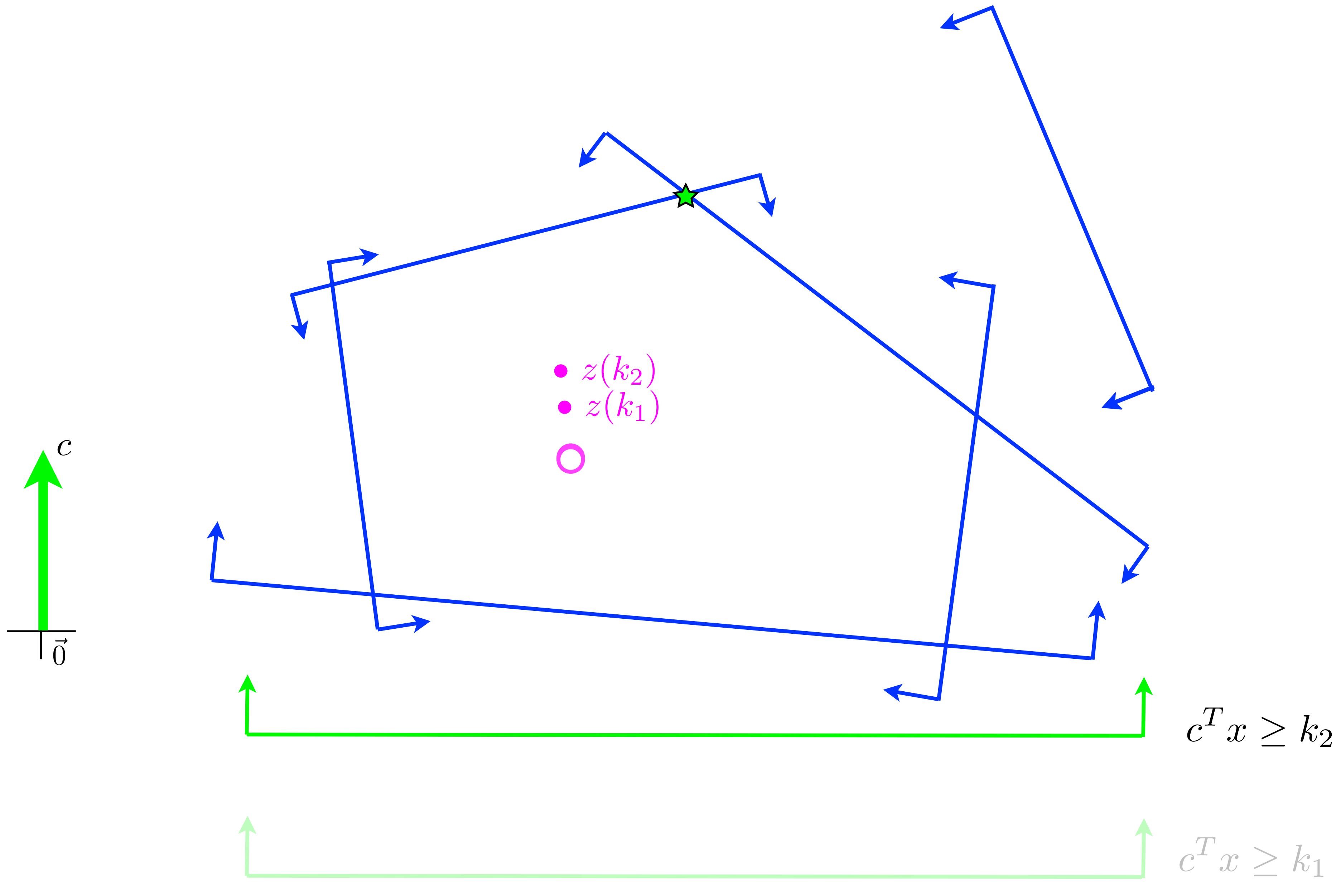




Add a new constraint, perpendicular to c: $\quad c^T x \geq k_1$ for some constant k_1

This changes the analytic center, the equilibrium point.

Denote the new analytic center by $z(k_1)$.

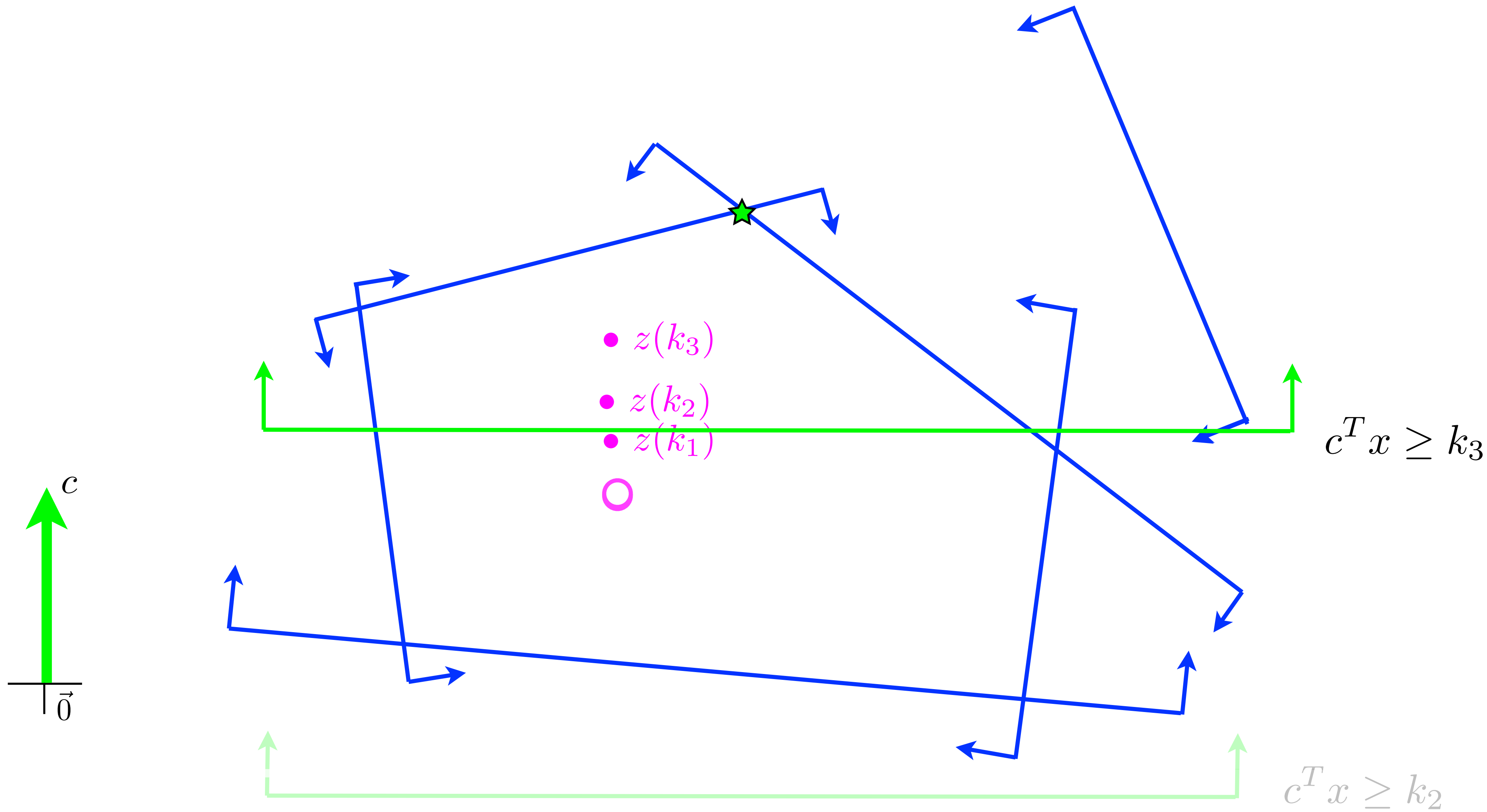


Add a new constraint, perpendicular to c: $\quad c^T x \geq k_1$ for some constant k_1

This changes the analytic center, the equilibrium point.

Denote the new analytic center by $z(k_1)$.

Now increase k_1 to a new constant k_2, giving a new analytic center $z(k_2)$.



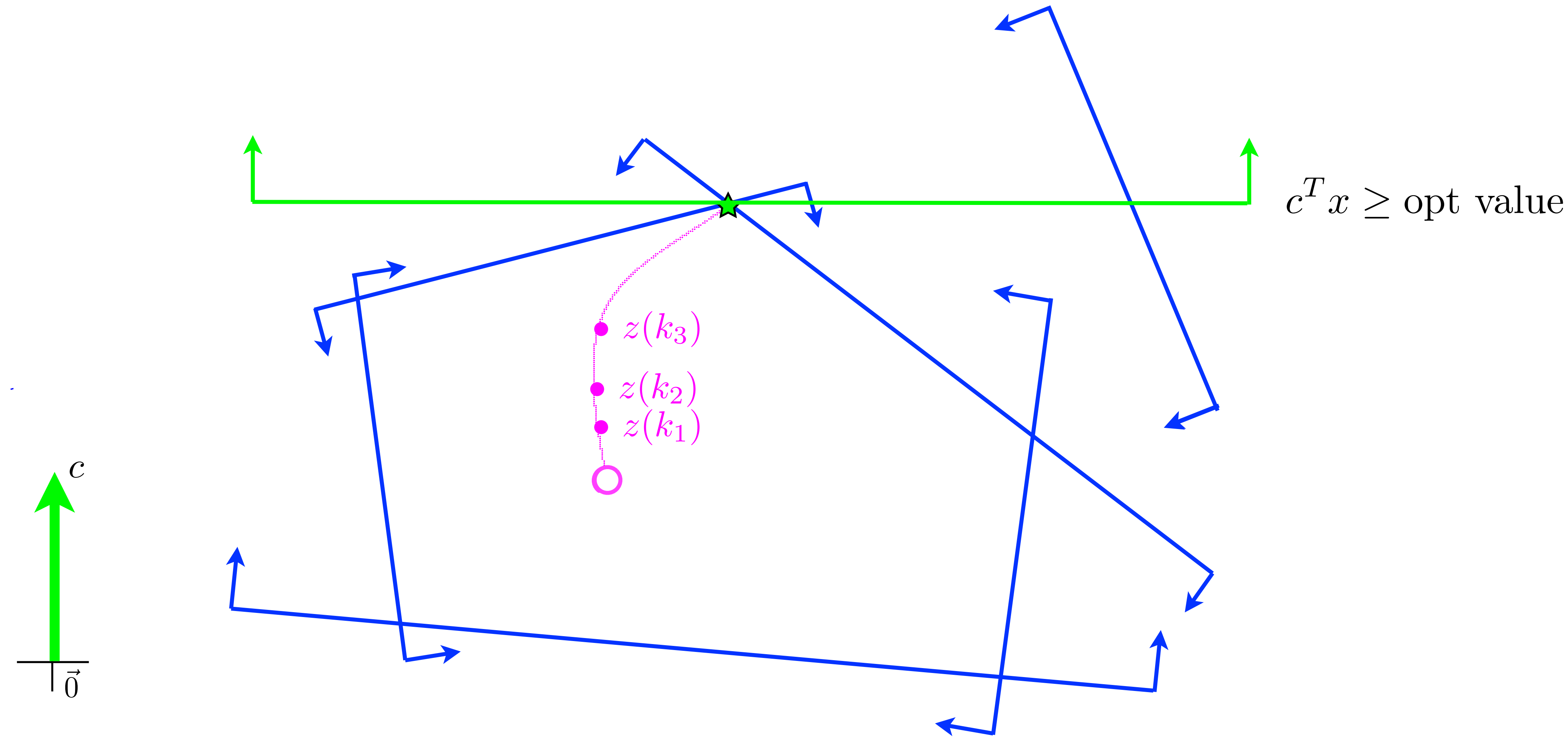
Add a new constraint, perpendicular to c: $c^T x \geq k_1$ for some constant k_1

This changes the analytic center, the equilibrium point.

Denote the new analytic center by $z(k_1)$.

Now increase k_1 to a new constant k_2, giving a new analytic center $z(k_2)$.

And so on: $z(k_1), z(k_2), z(k_3), \ldots$



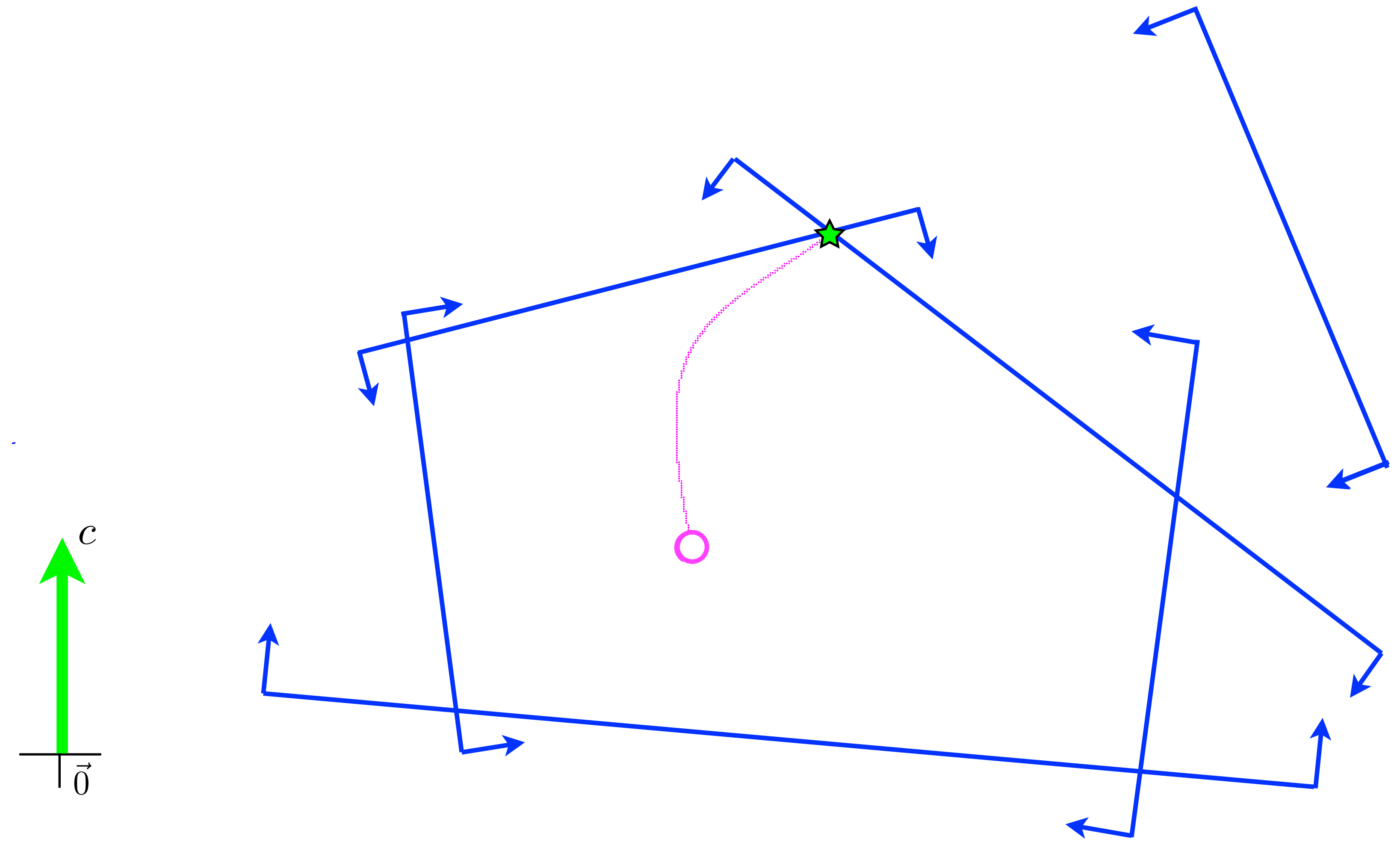
Add a new constraint, perpendicular to c: $c^T x \geq k_1$ for some constant k_1

This changes the analytic center, the equilibrium point.

Denote the new analytic center by $z(k_1)$.

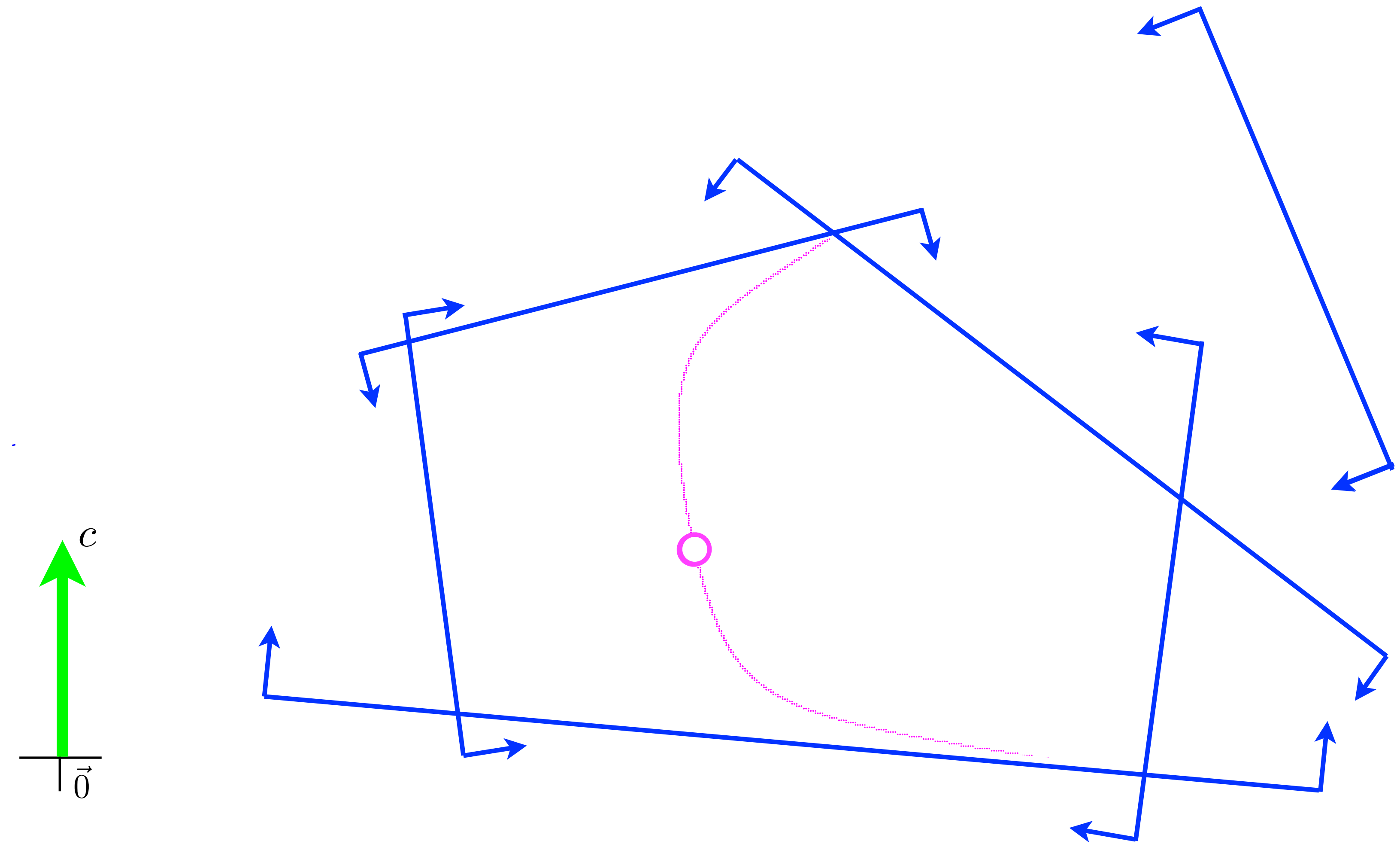
Now increase k_1 to a new constant k_2, giving a new analytic center $z(k_2)$.

And so on: $z(k_1), z(k_2), z(k_3), \ldots$



the "central path"

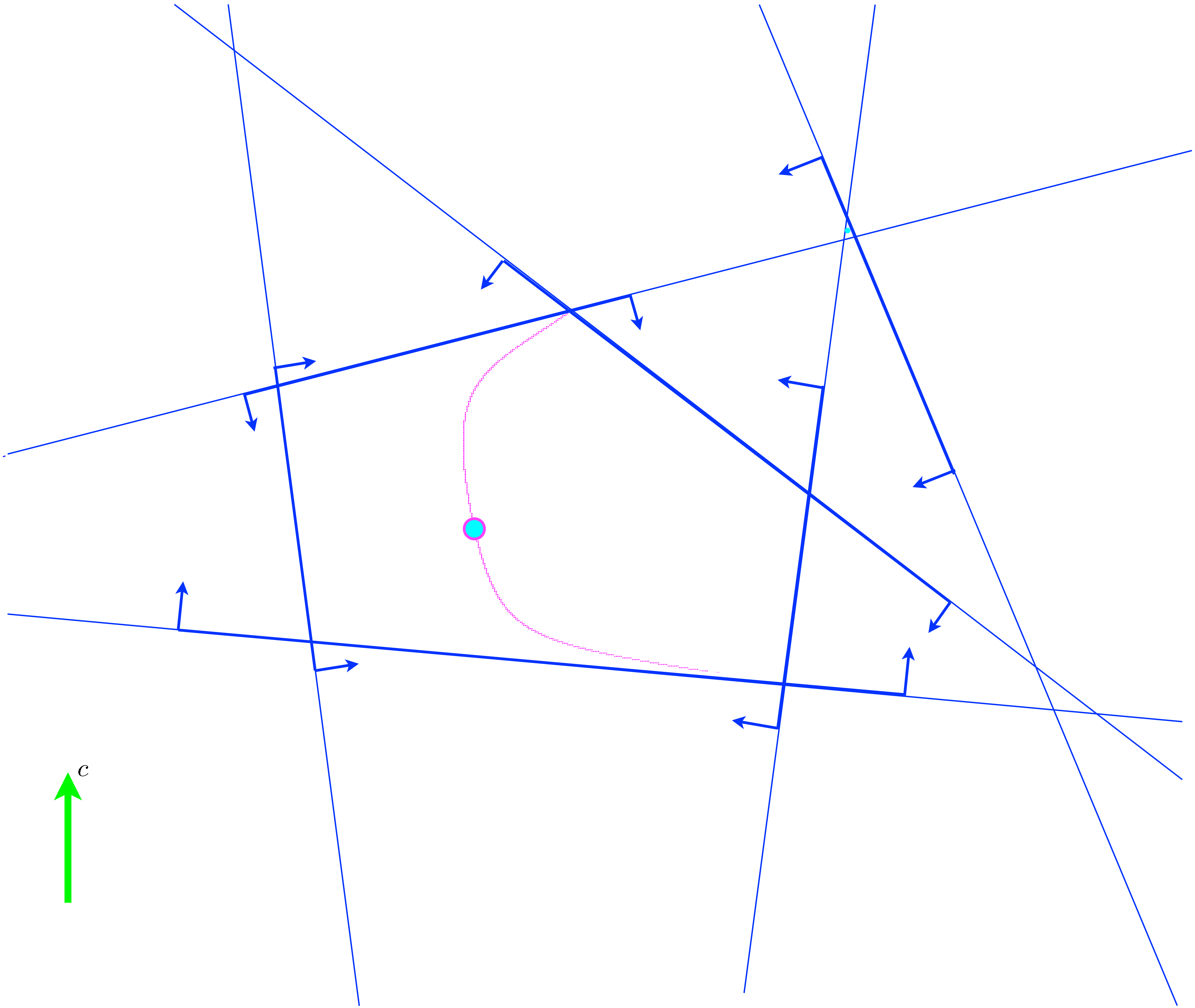
(a.k.a., "the path of analytic centers")



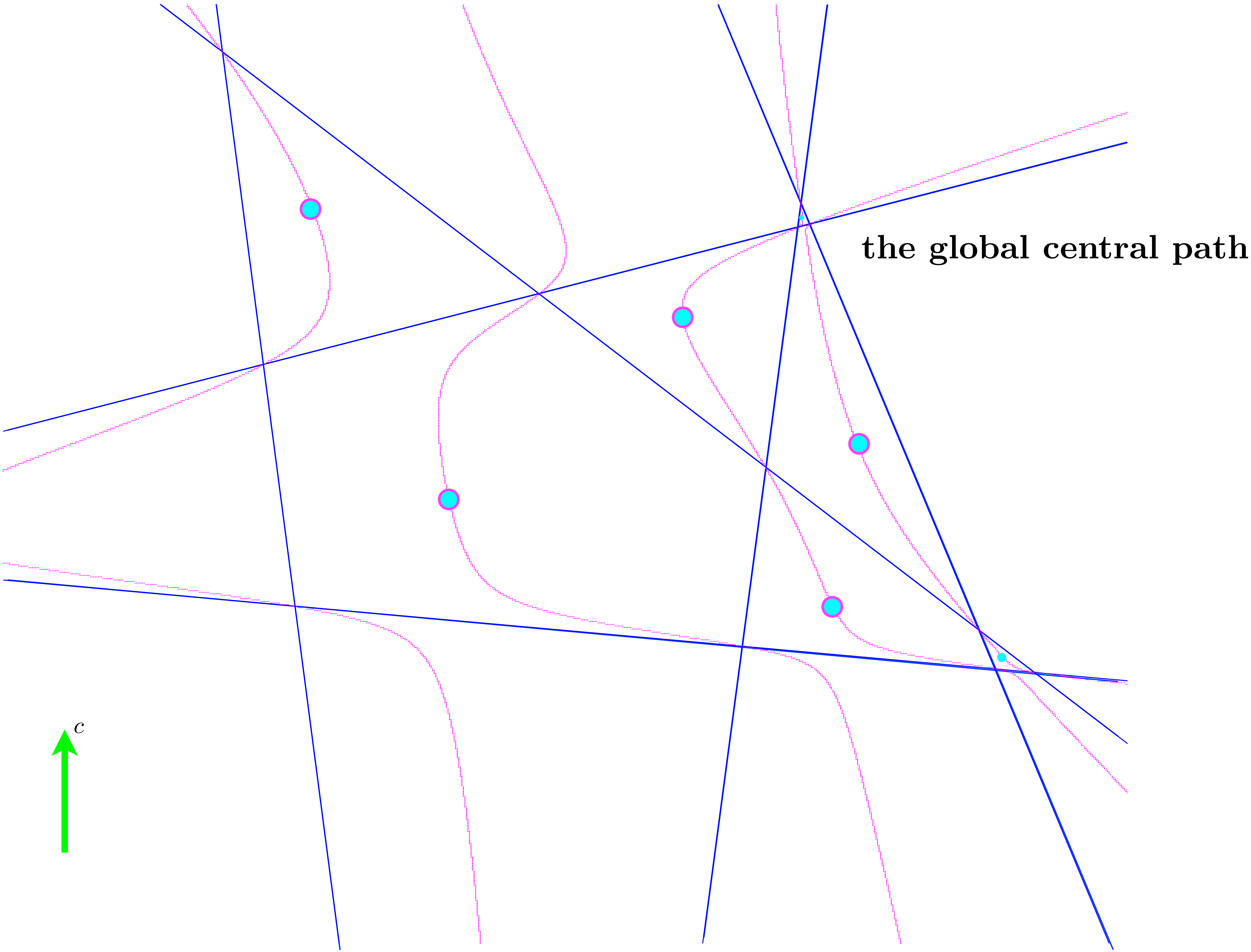
the "central path"

(a.k.a., "the path of analytic centers")

If we do likewise with c replaced by $-c$,
the path extends to the feasible point minimizing $c^T x$.

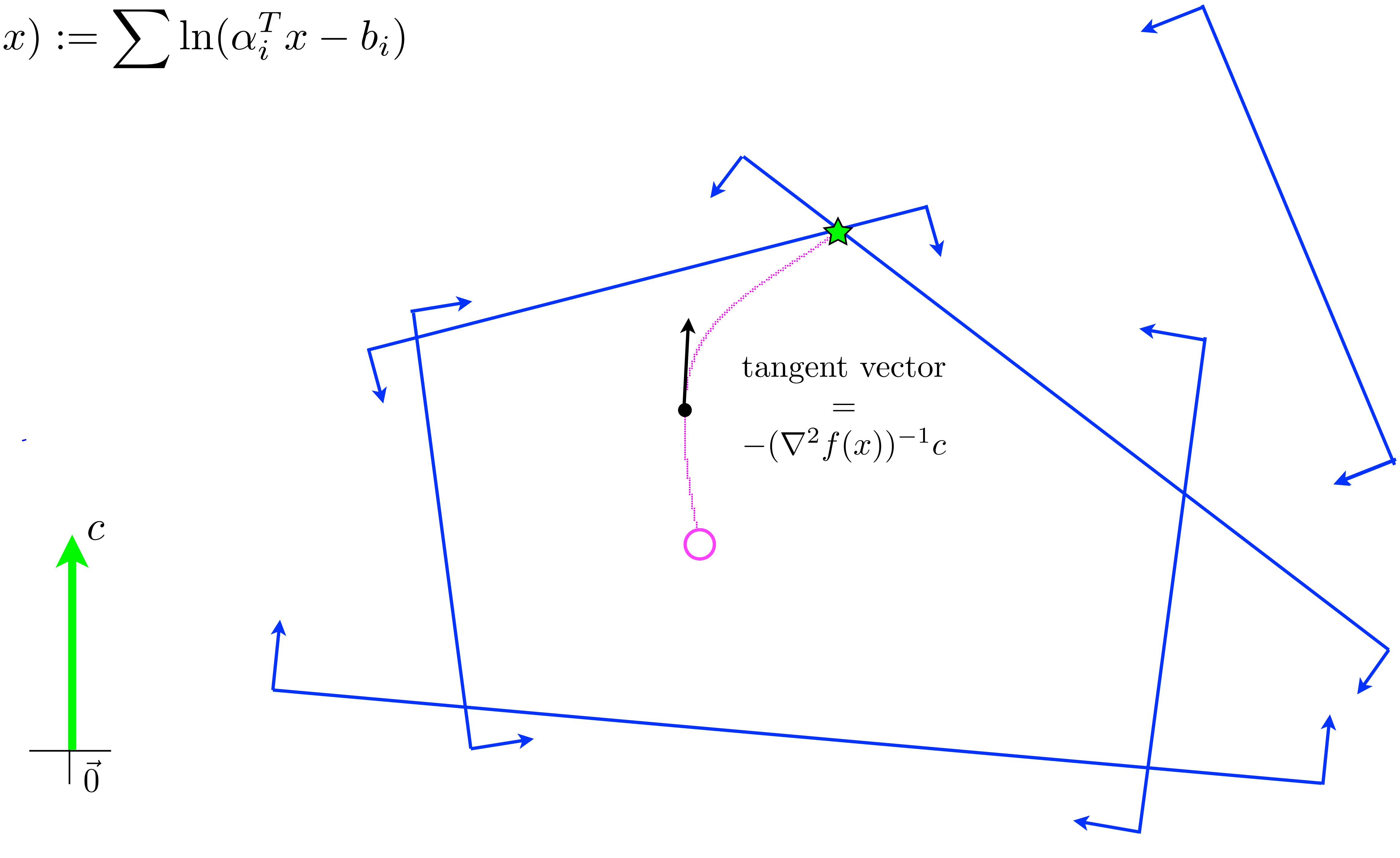


c

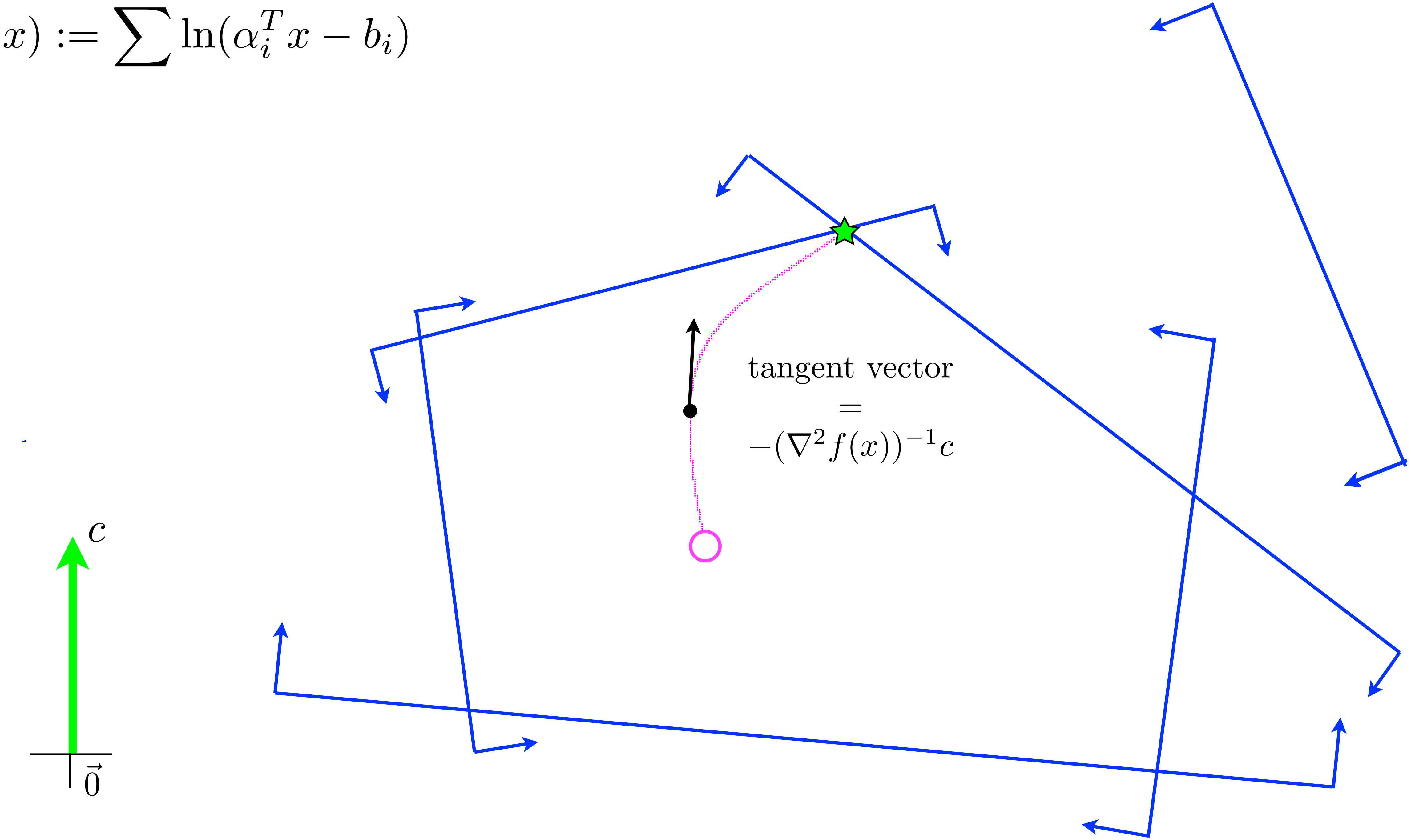


the global central path
c

$$f(x) := \sum \ln(\alpha_i^T x - b_i)$$



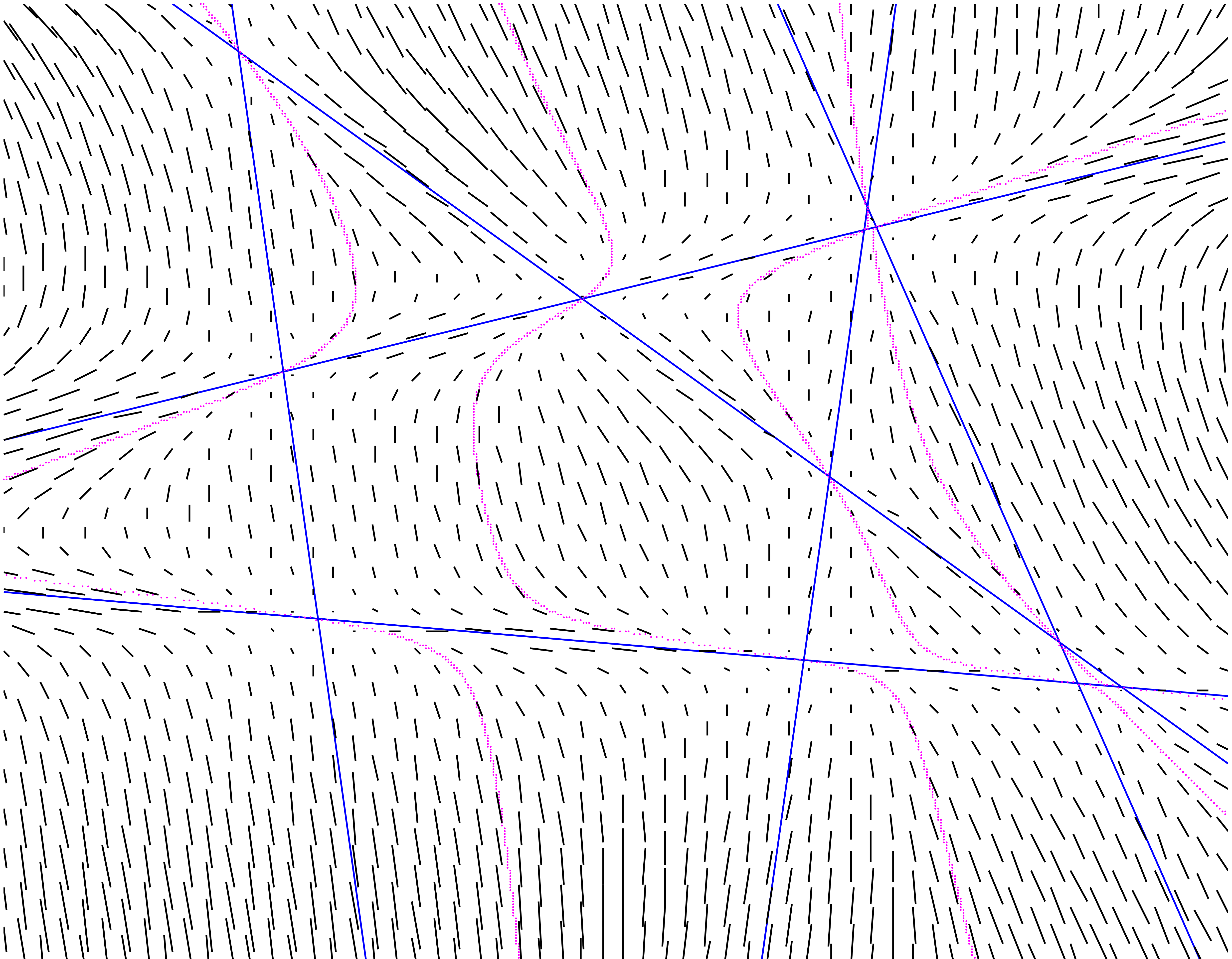
$f(x) := \sum \ln(\alpha_i^T x - b_i)$



Natural to consider vector field $x \mapsto -(\nabla^2 f(x))^{-1}c$ for all interior x

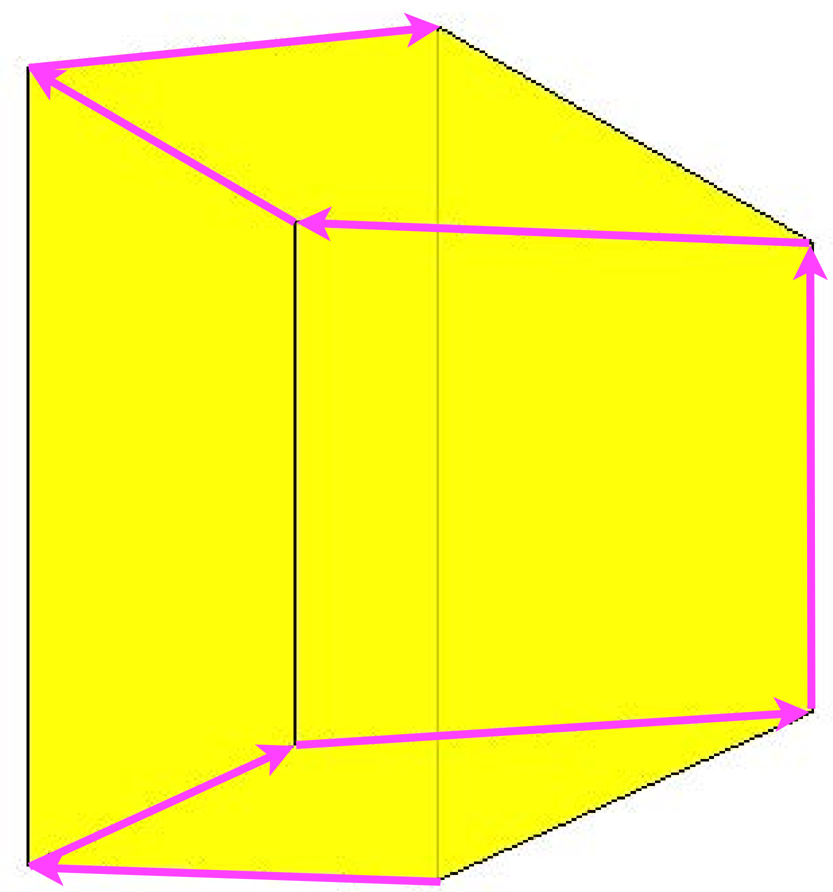
(not just for x on the central path)

– this yields the "affine scaling algorithm"



A theorem in Mike's first optimization paper (1985-86, with N. Megiddo):

When initiated at appropriate interior points,
the affine-scaling flow closely follows
the bad path of the Klee-Minty cube.

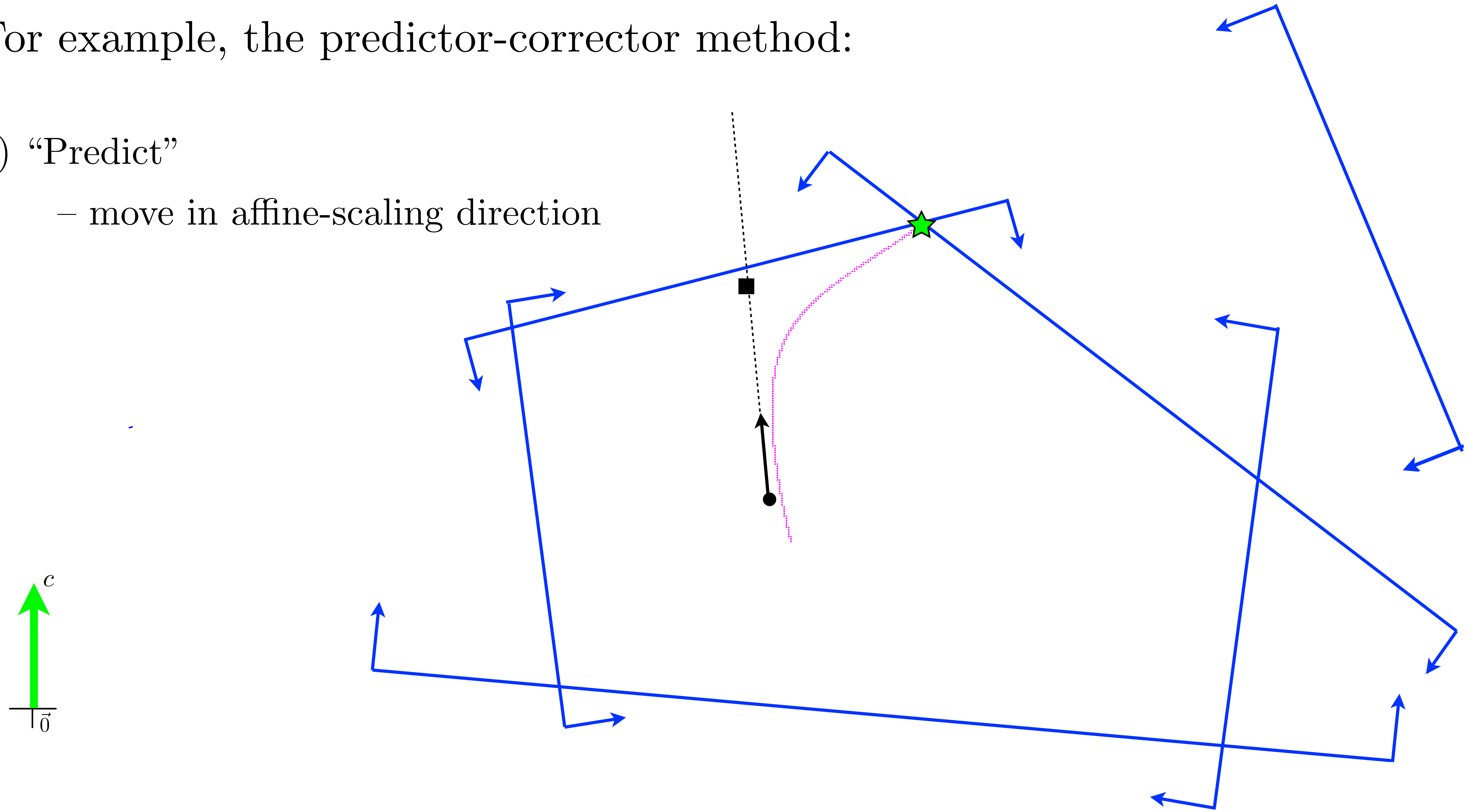


Mike's result (and subsequent results) indicate the importance for algorithms to remain near the central path in order to acheive polynomial time.

For example, the predictor-corrector method:

1) "Predict"

– move in affine-scaling direction



Mike's result (and subsequent results) indicate the importance for algorithms to remain near the central path in order to acheive polynomial time.

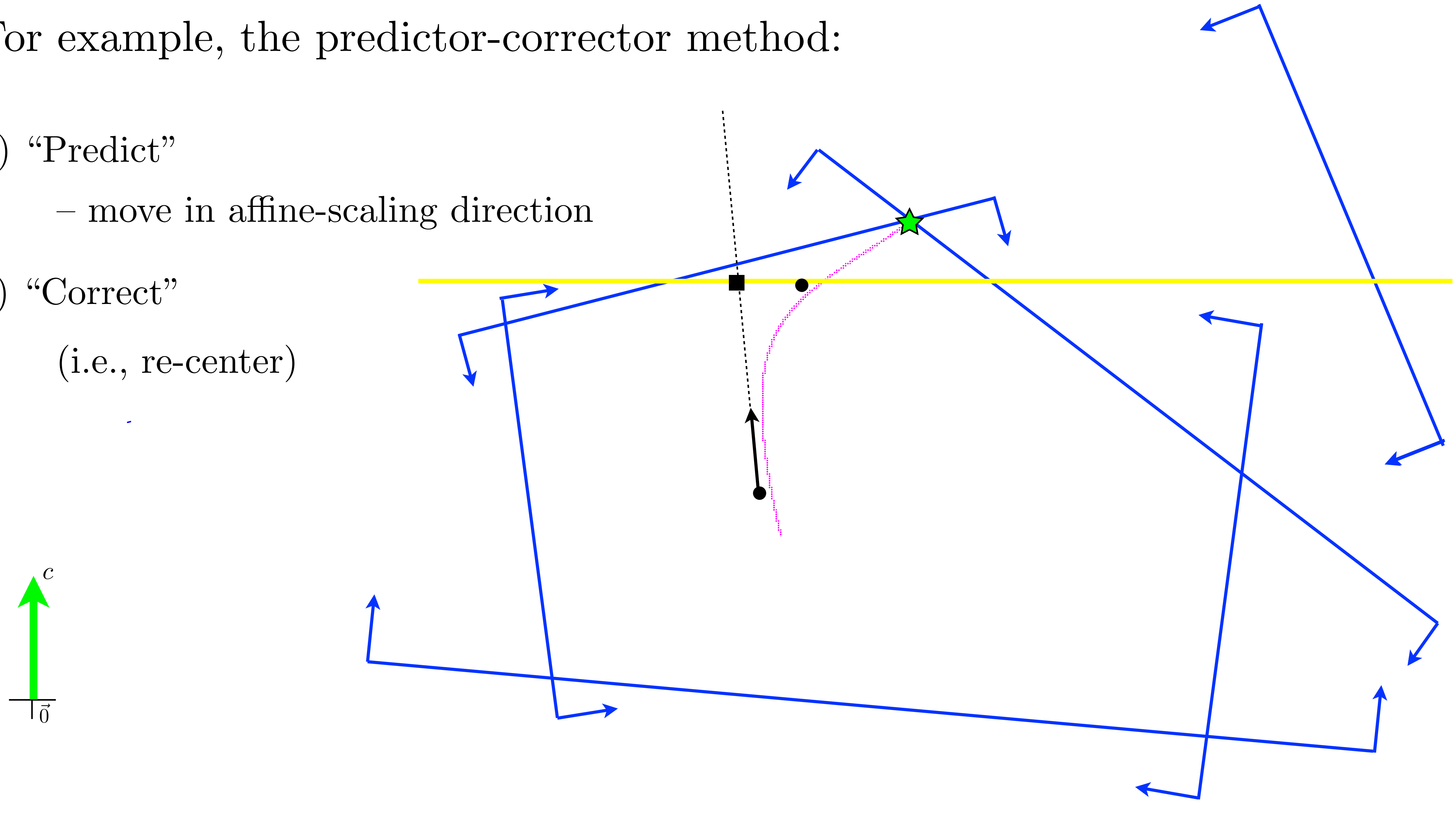
For example, the predictor-corrector method:

1) "Predict"

– move in affine-scaling direction

2) "Correct"

(i.e., re-center)



Mike's result (and subsequent results) indicate the importance for algorithms to remain near the central path in order to acheive polynomial time.

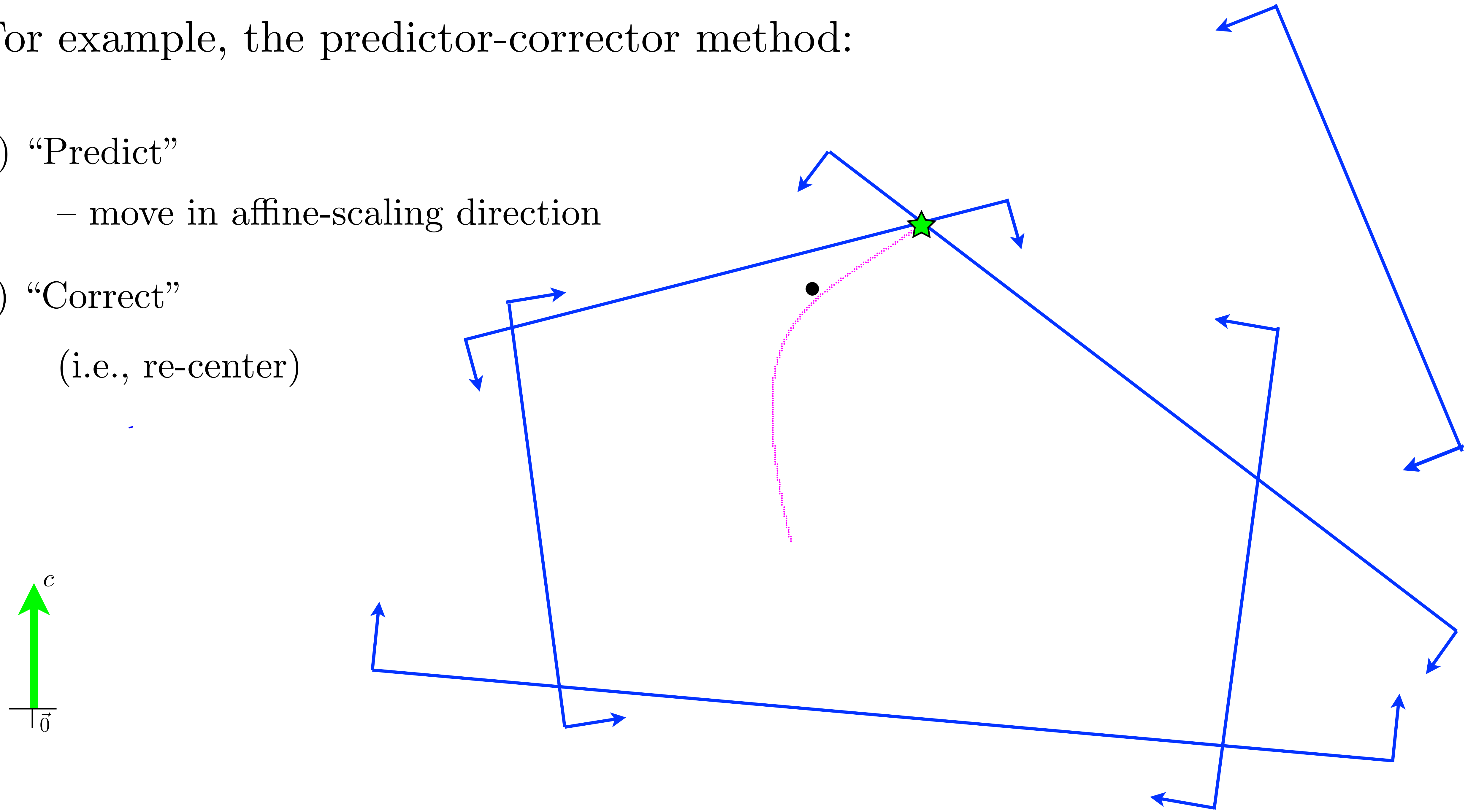
For example, the predictor-corrector method:

1) "Predict"

– move in affine-scaling direction

2) "Correct"

(i.e., re-center)



Mike's result (and subsequent results) indicate the importance for algorithms to remain near the central path in order to acheive polynomial time.

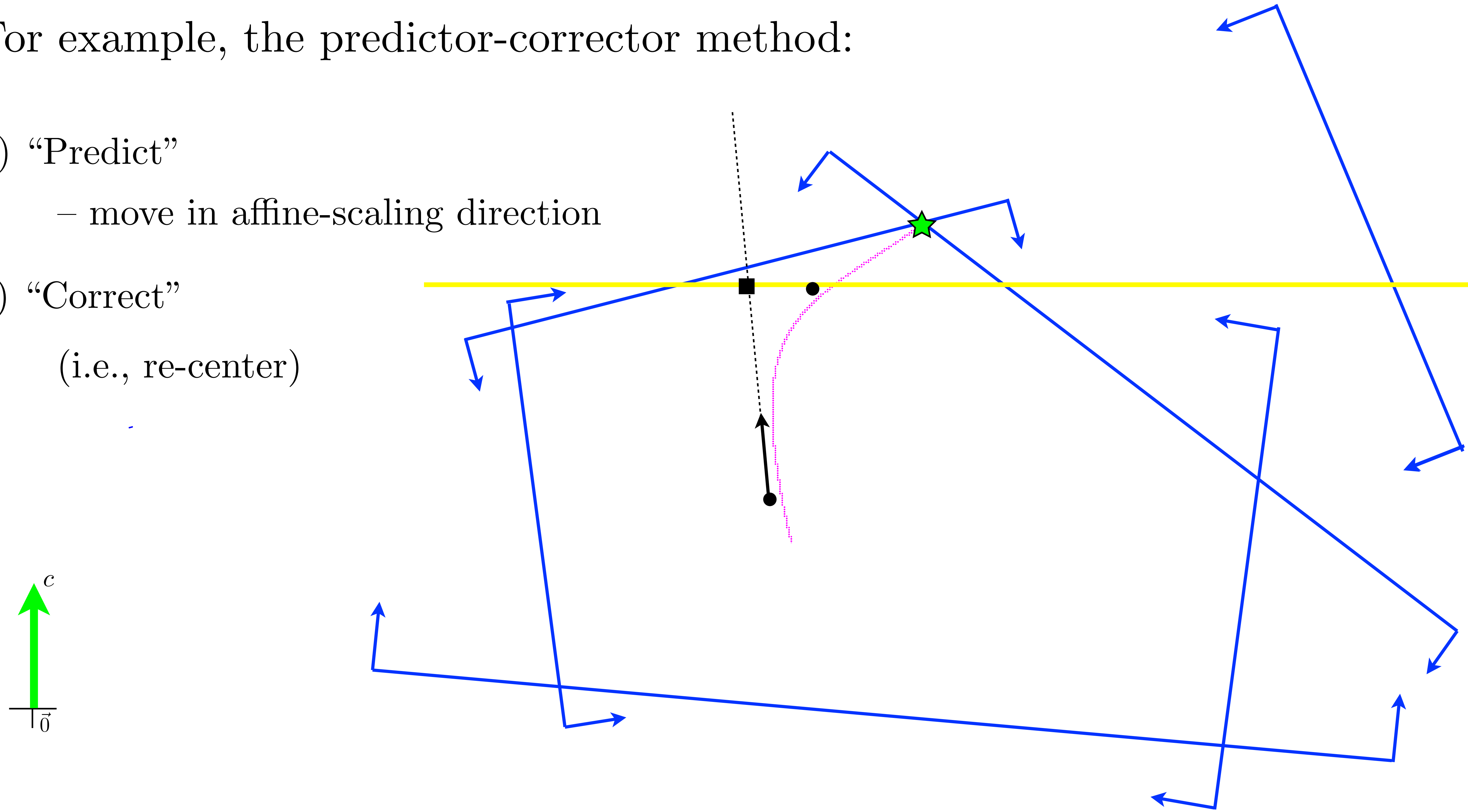
For example, the predictor-corrector method:

1) "Predict"

– move in affine-scaling direction

2) "Correct"

(i.e., re-center)



The straighter the central path, the better!

Duality & Dynamics

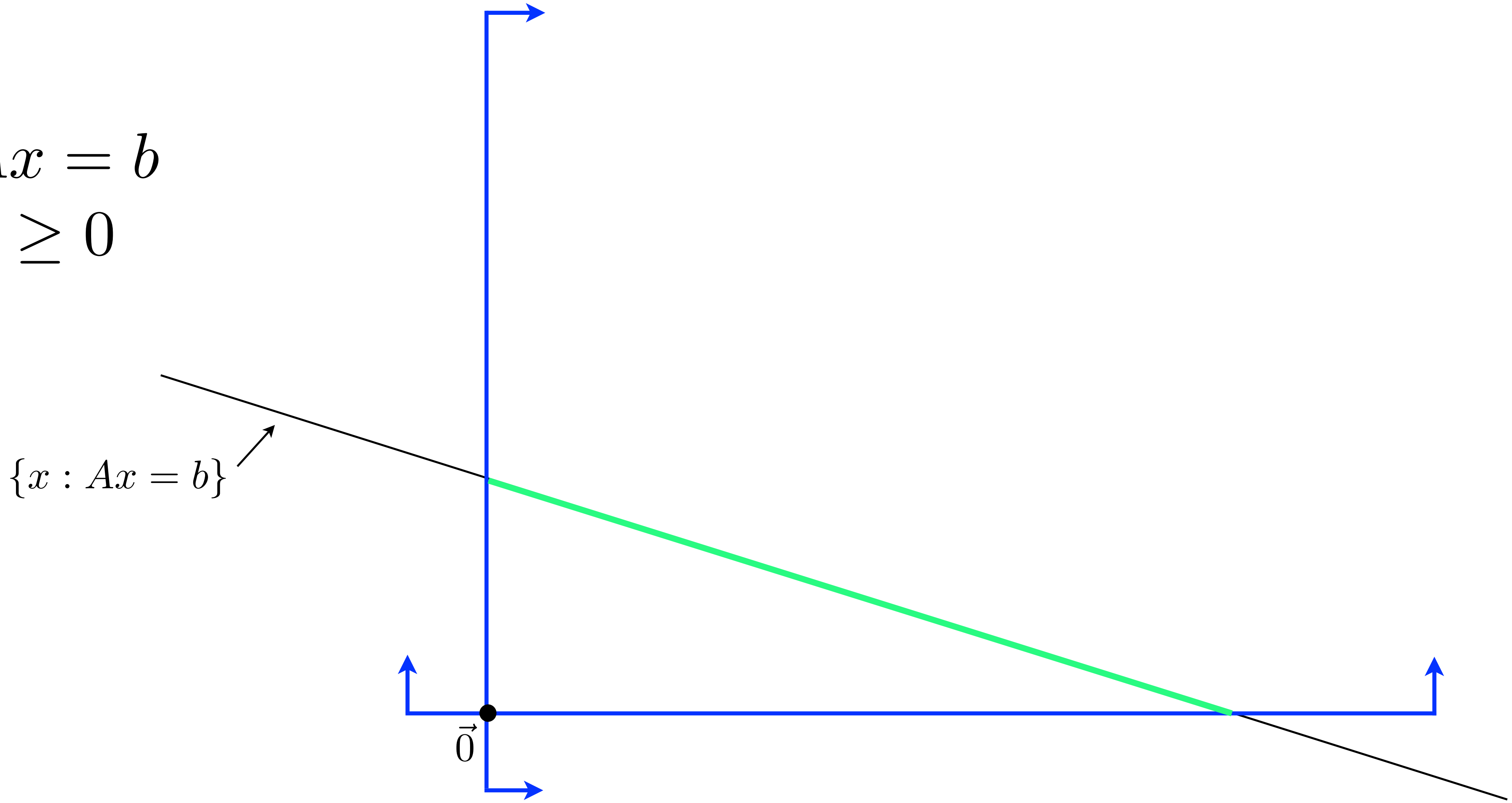
Many contributors, all the way back to:

N. Megiddo, "Pathways to the optimal set in linear programming" (1988)

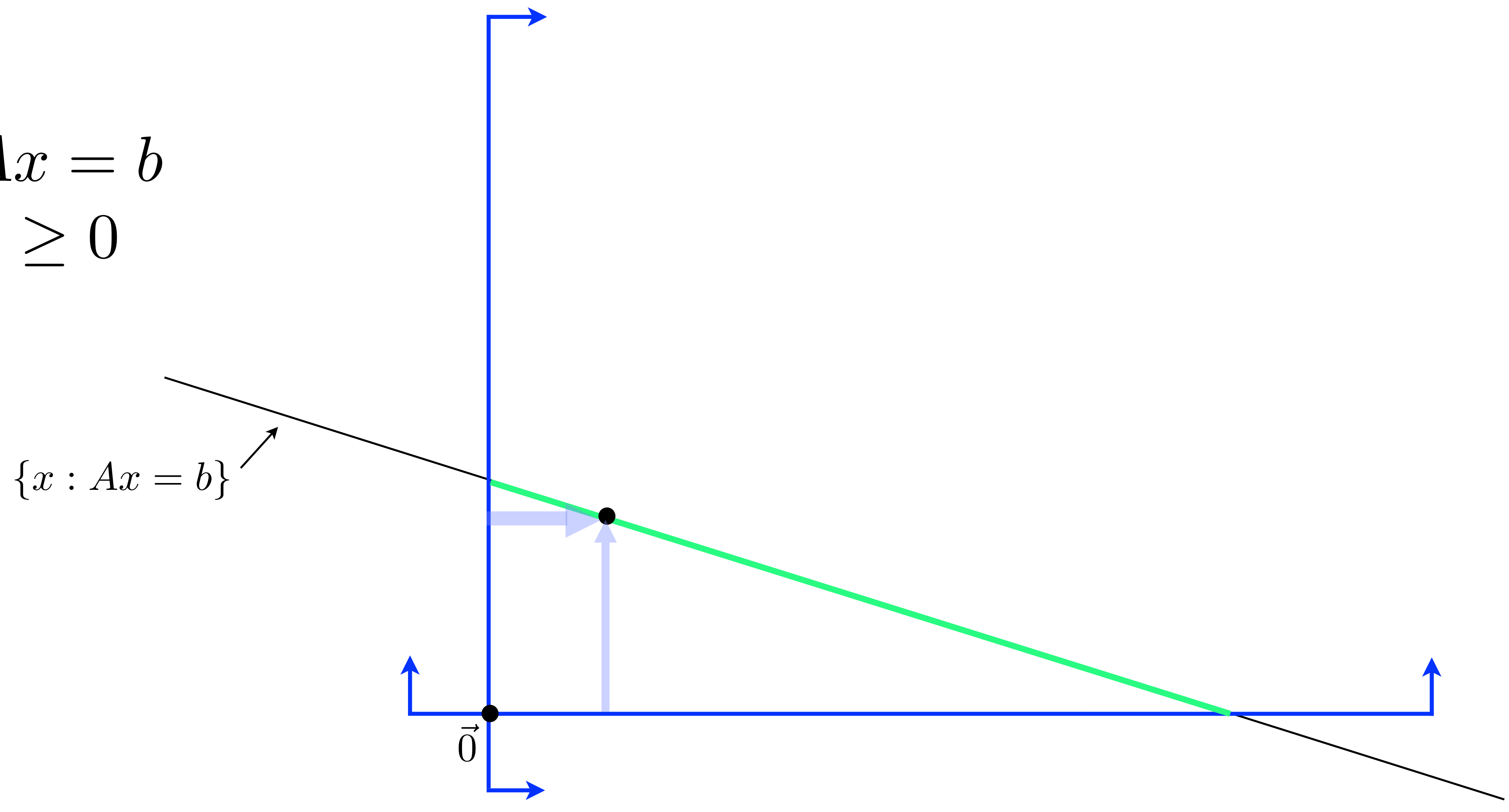
M. Kojima, S. Mizuno, and A. Yoshise,
"A primal-dual interior point algorithm for linear programming" (1988)

R.D.C. Monteiro, I. Adler,
"Interior path following primal-dual algorithms" (1989)

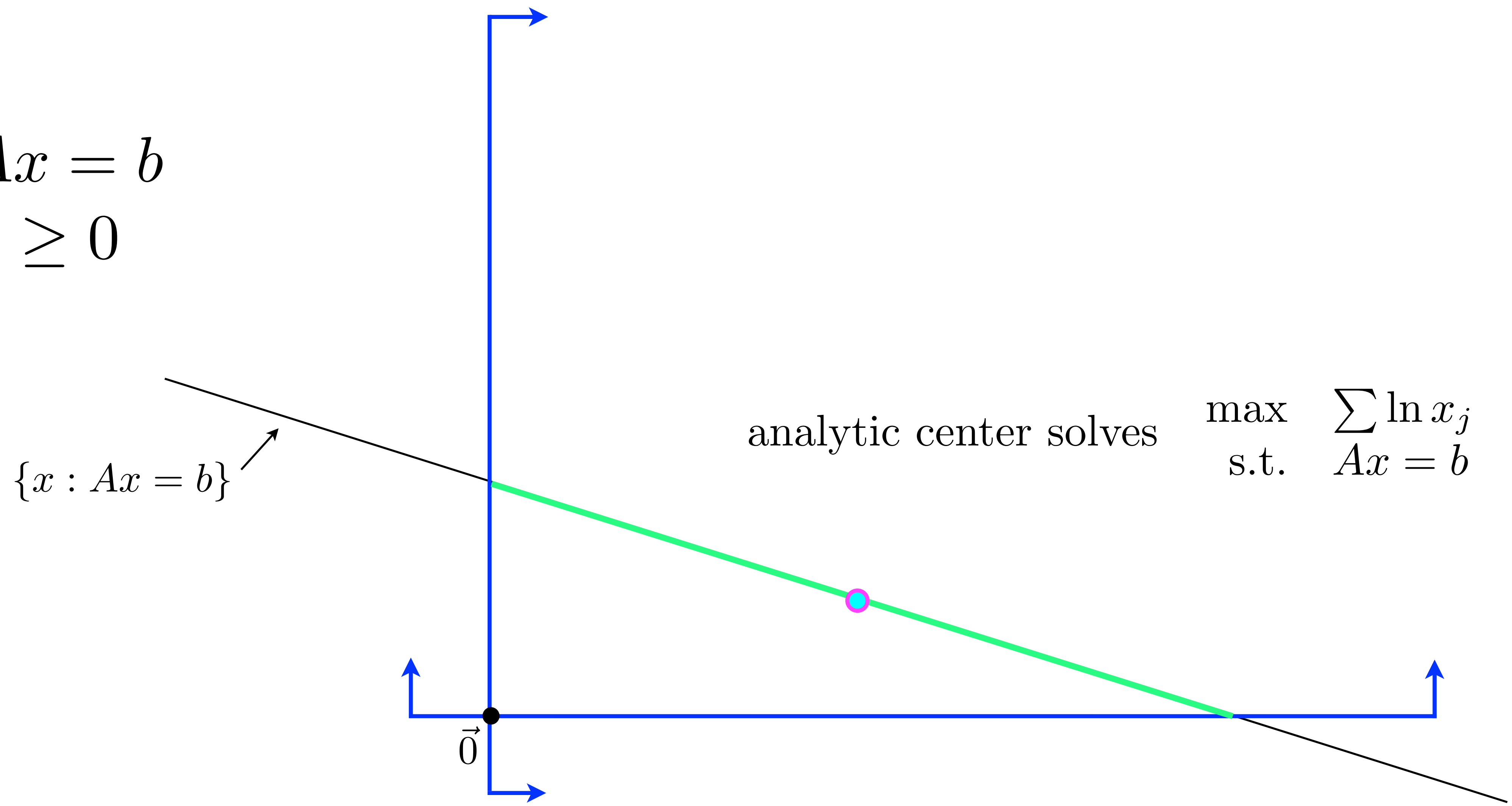
$$Ax = b$$
$$x \geq 0$$



$$Ax = b$$
$$x \geq 0$$

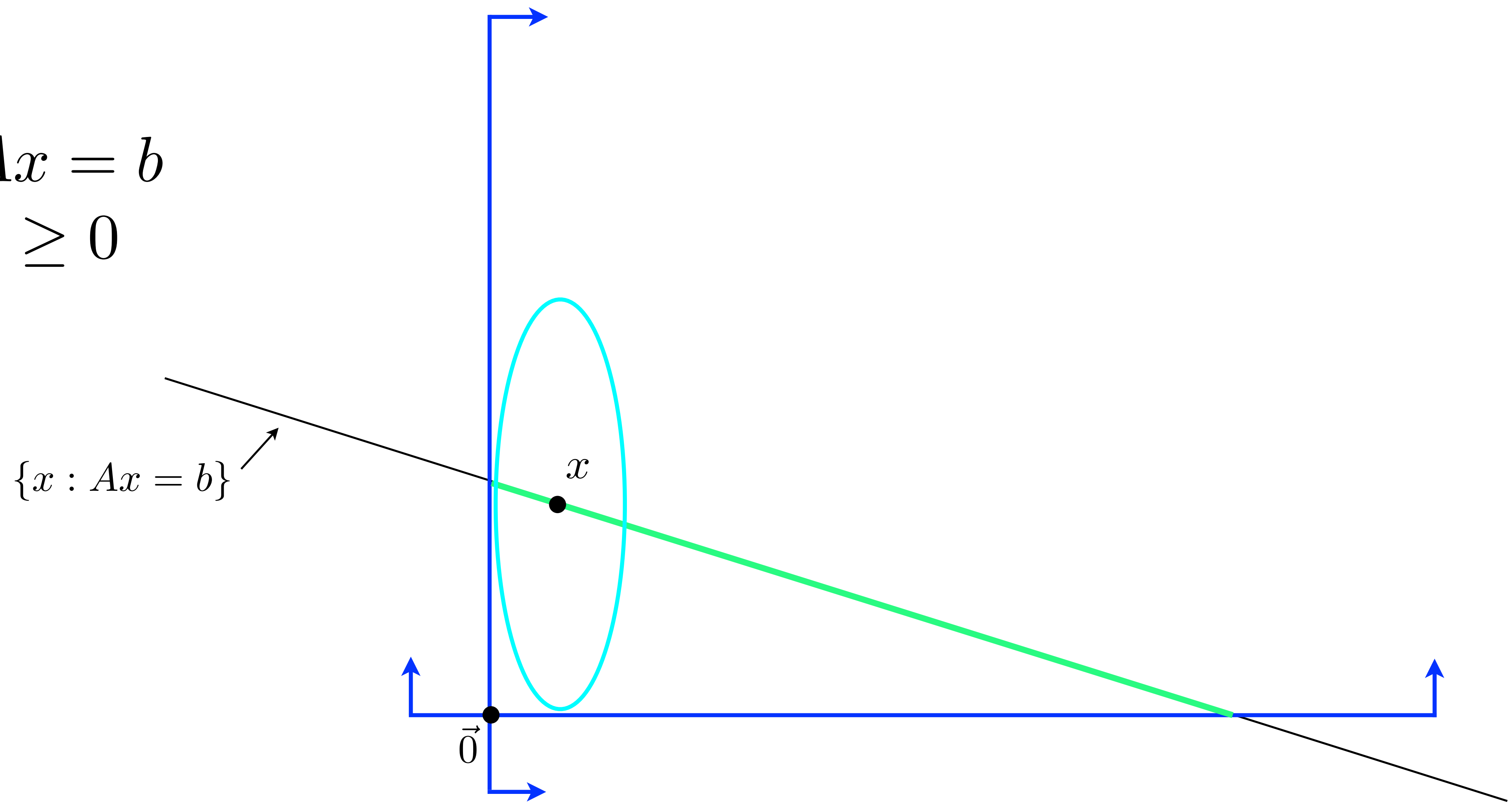


$$Ax = b$$
$$x \geq 0$$



analytic center solves $\begin{array}{rl} \max & \sum \ln x_j \\ \text{s.t.} & Ax = b \end{array}$

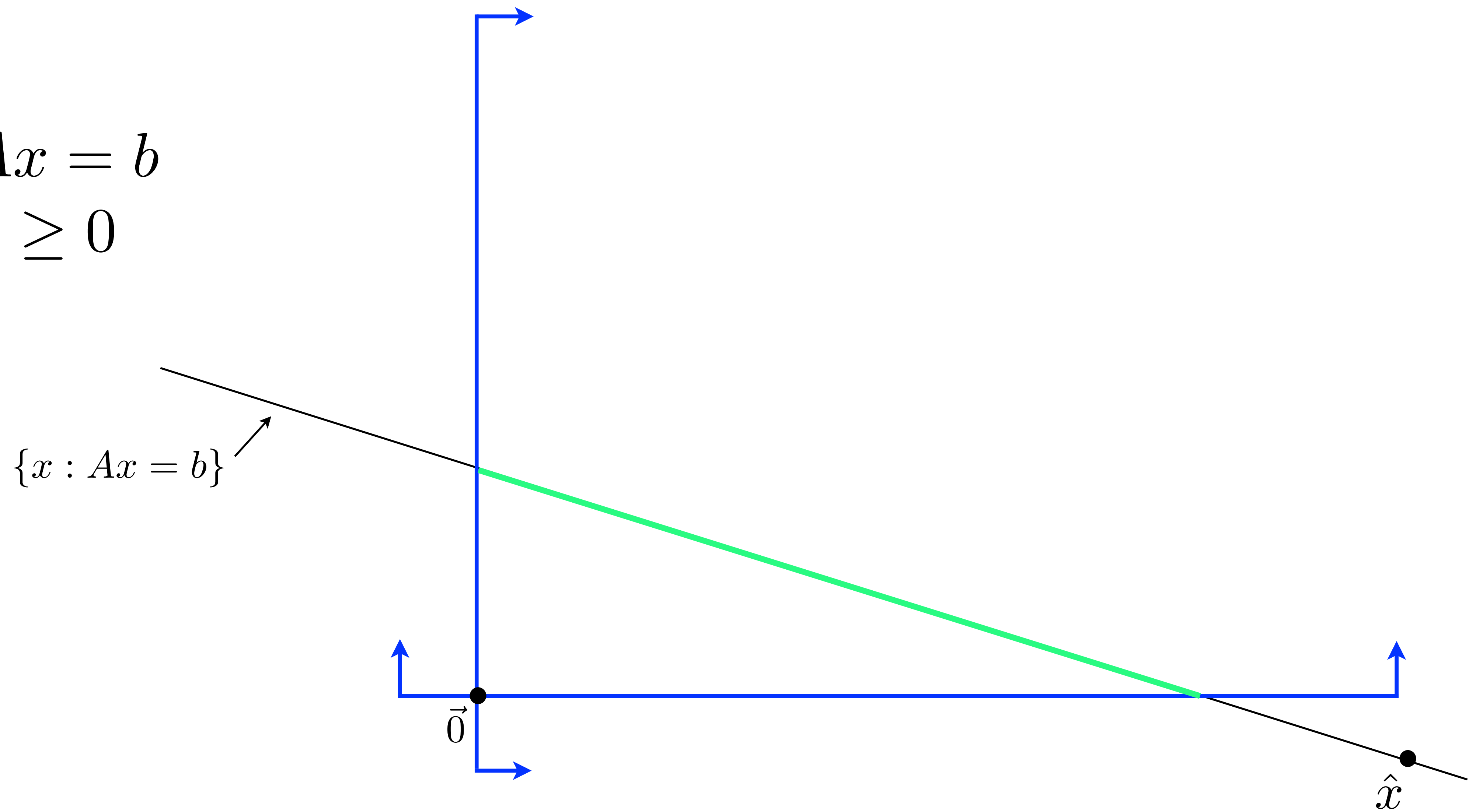
$$Ax = b$$
$$x \geq 0$$



The relevant inner product at interior x is

$$\langle u, \hat{u} \rangle_x = \sum \frac{u_j \hat{u}_j}{x_j^2}$$

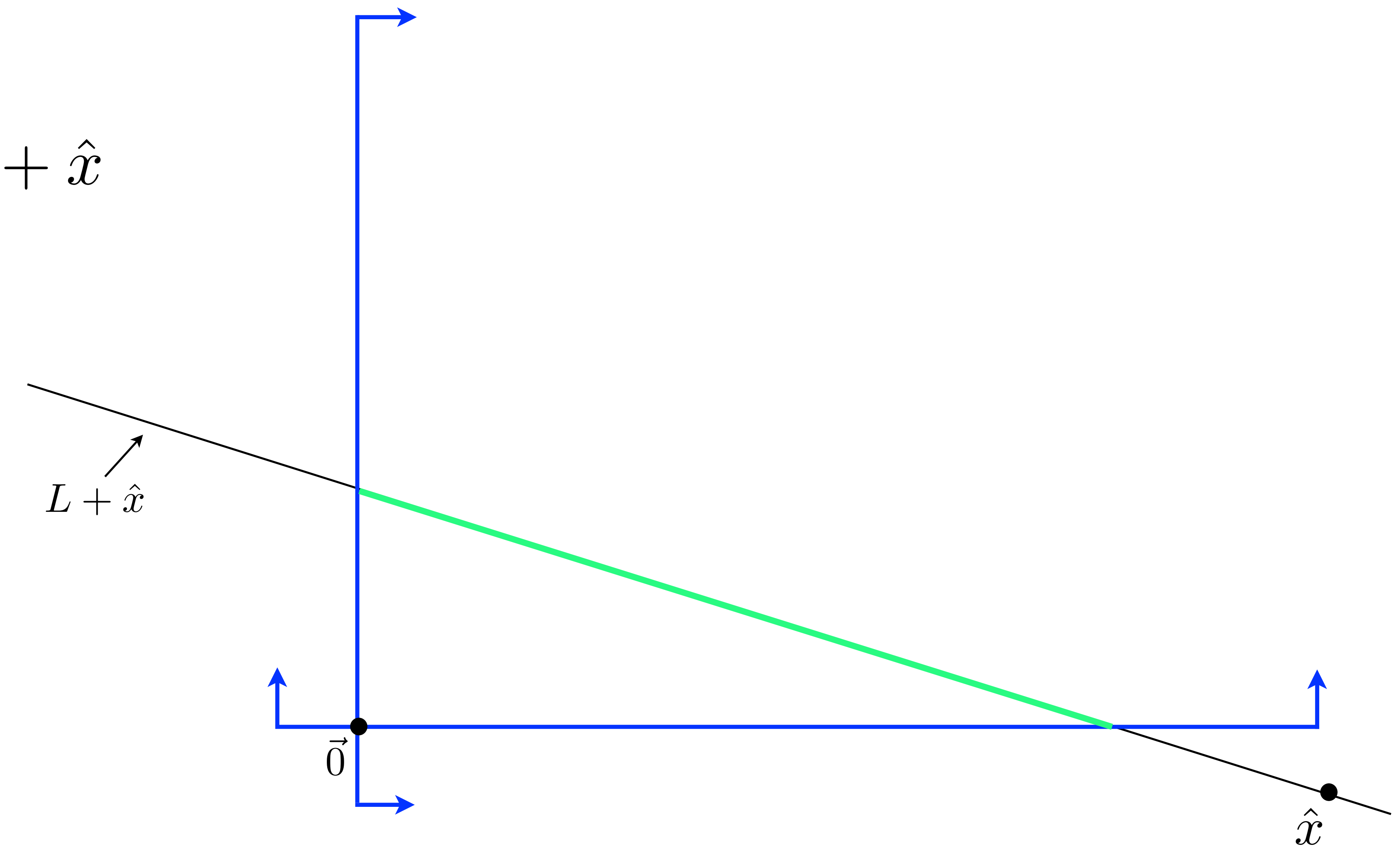
$$Ax = b$$
$$x \geq 0$$



– fix $\hat{x}$ satisfying $Ax = b$
and let L be the nullspace of A

$x \in L + \hat{x}$

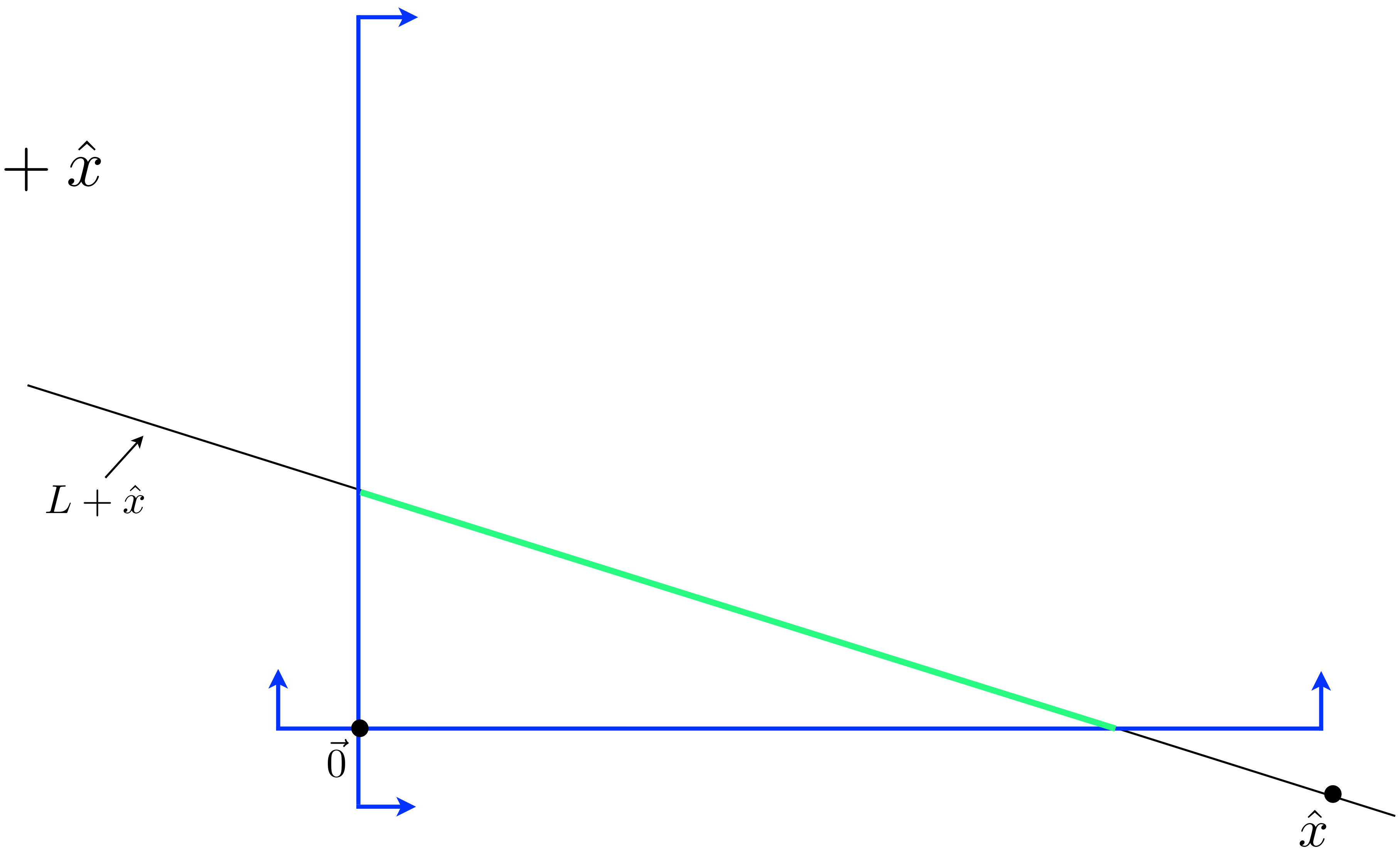
$x \geq 0$



– fix $\hat{x}$ satisfying $Ax = b$
and let L be the nullspace of A

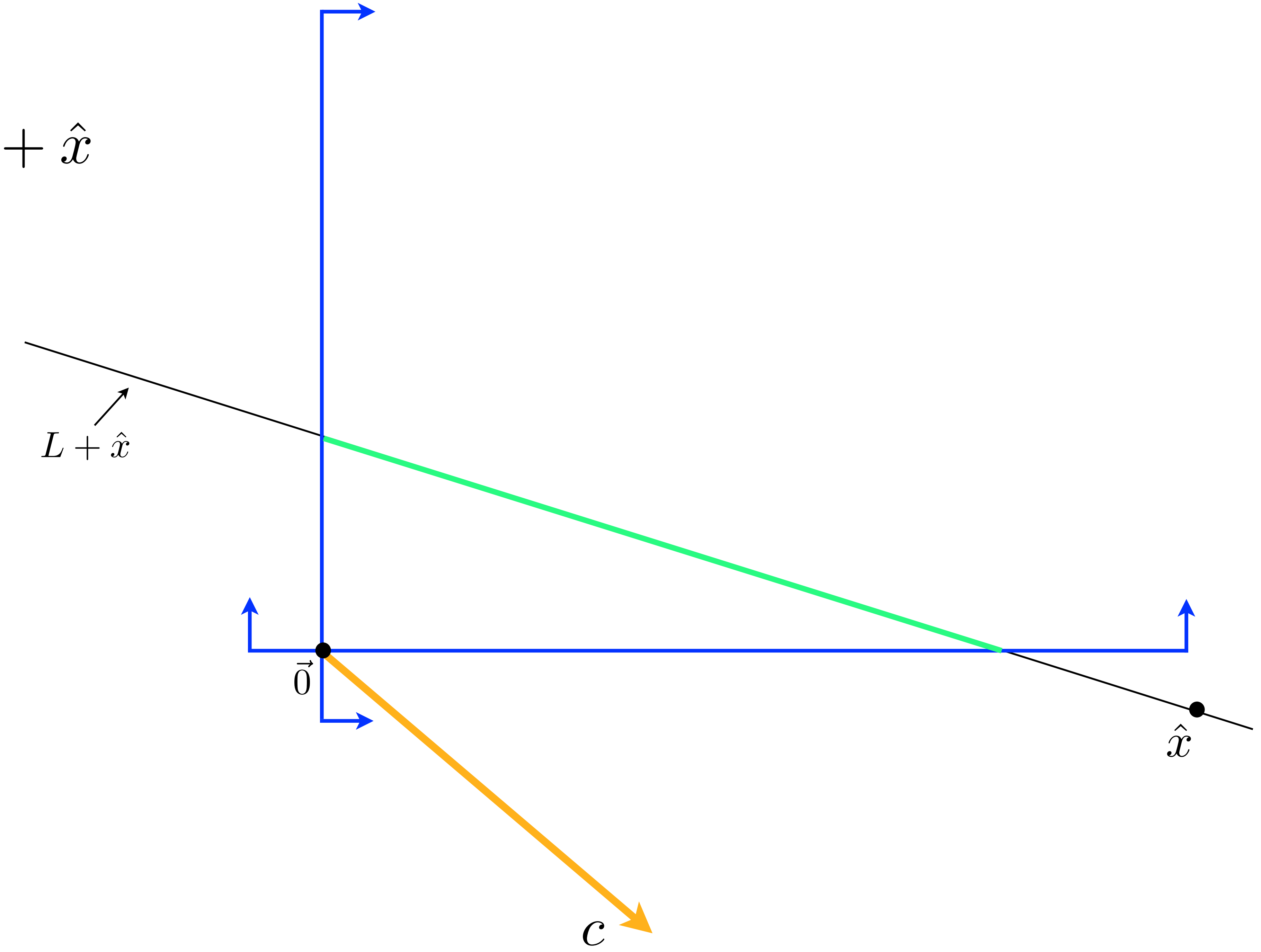
“min” rather than “max”

$$
\begin{aligned}
\min \quad & c^T x \\
\text{s.t.} \quad & x \in L + \hat{x} \\
& x \geq 0
\end{aligned}
$$



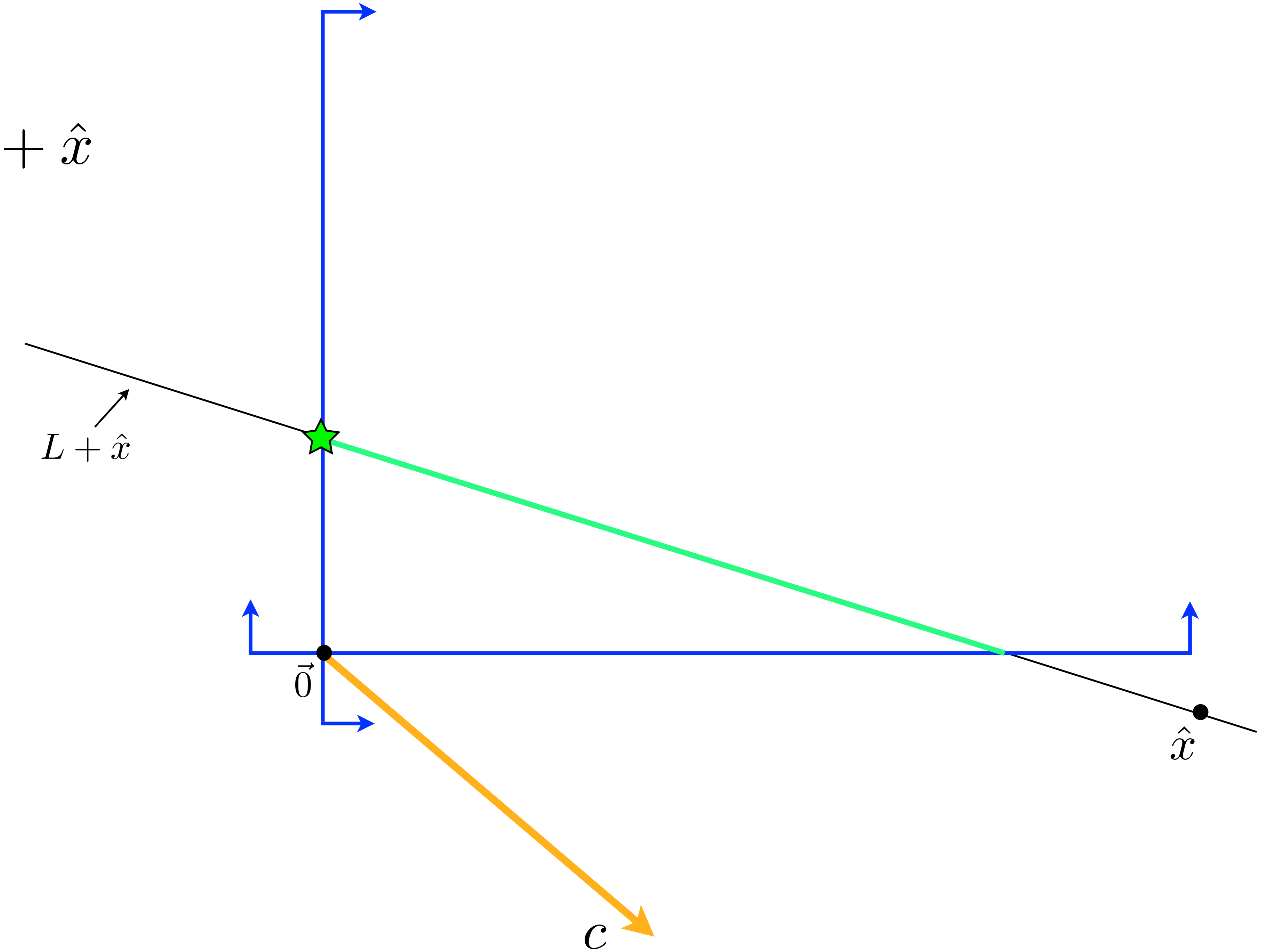
"min" rather than "max"

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & x \in L + \hat{x} \\ & x \geq 0 \end{aligned}$$



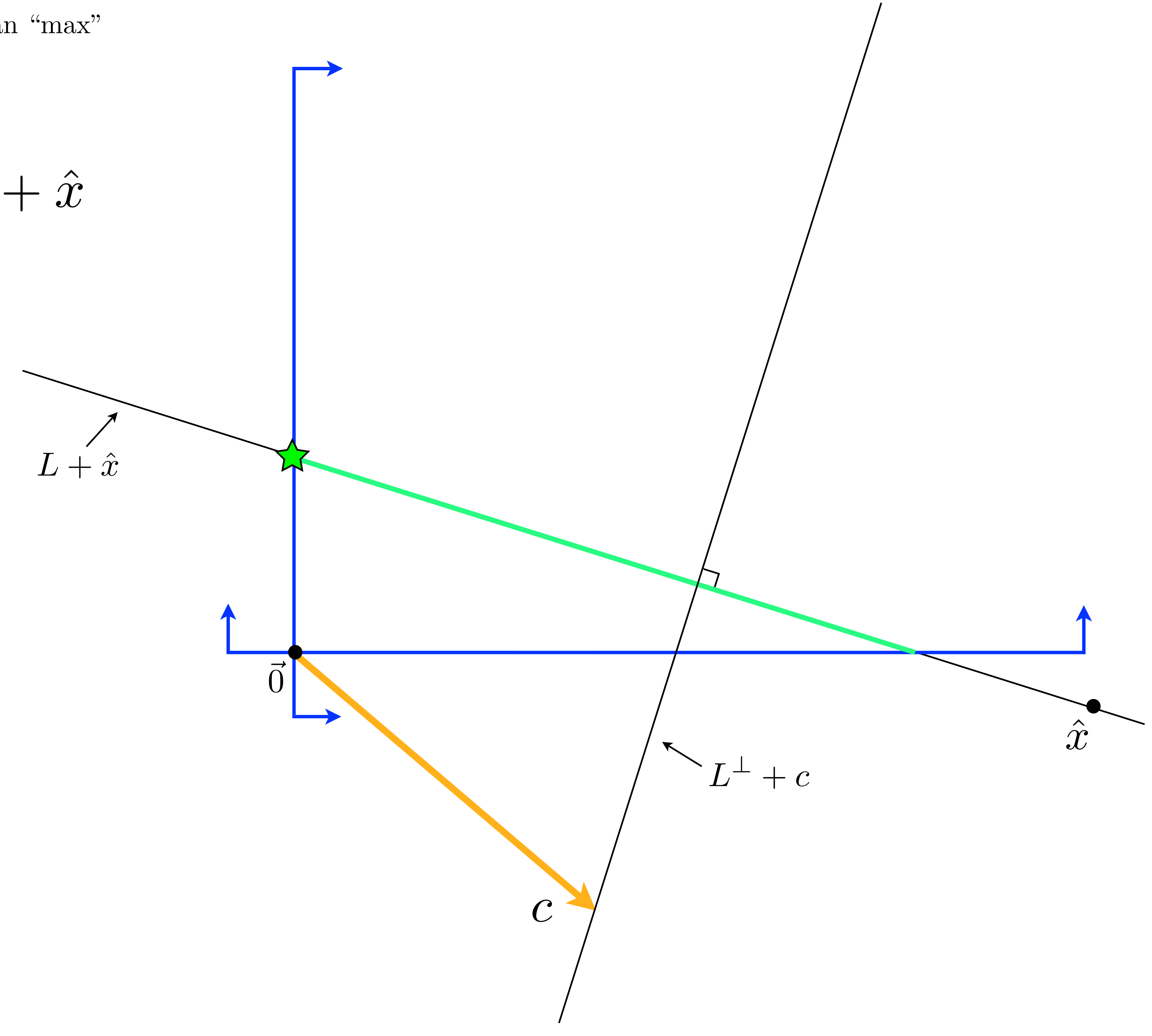
“min” rather than “max”

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & x \in L + \hat{x} \\ & x \geq 0 \end{aligned}$$



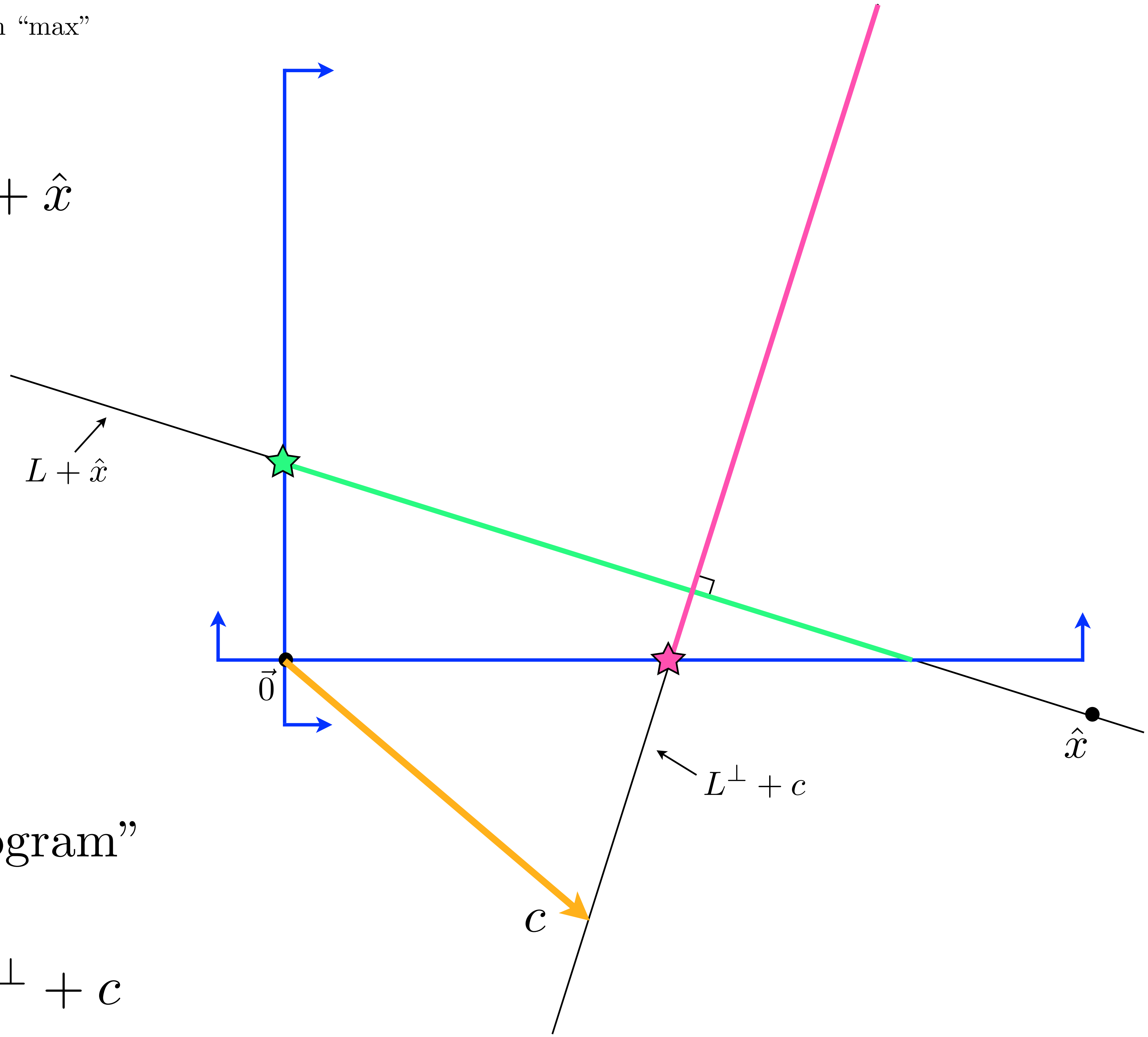
“min” rather than “max”

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & x \in L + \hat{x} \\ & x \geq 0 \end{array}$$



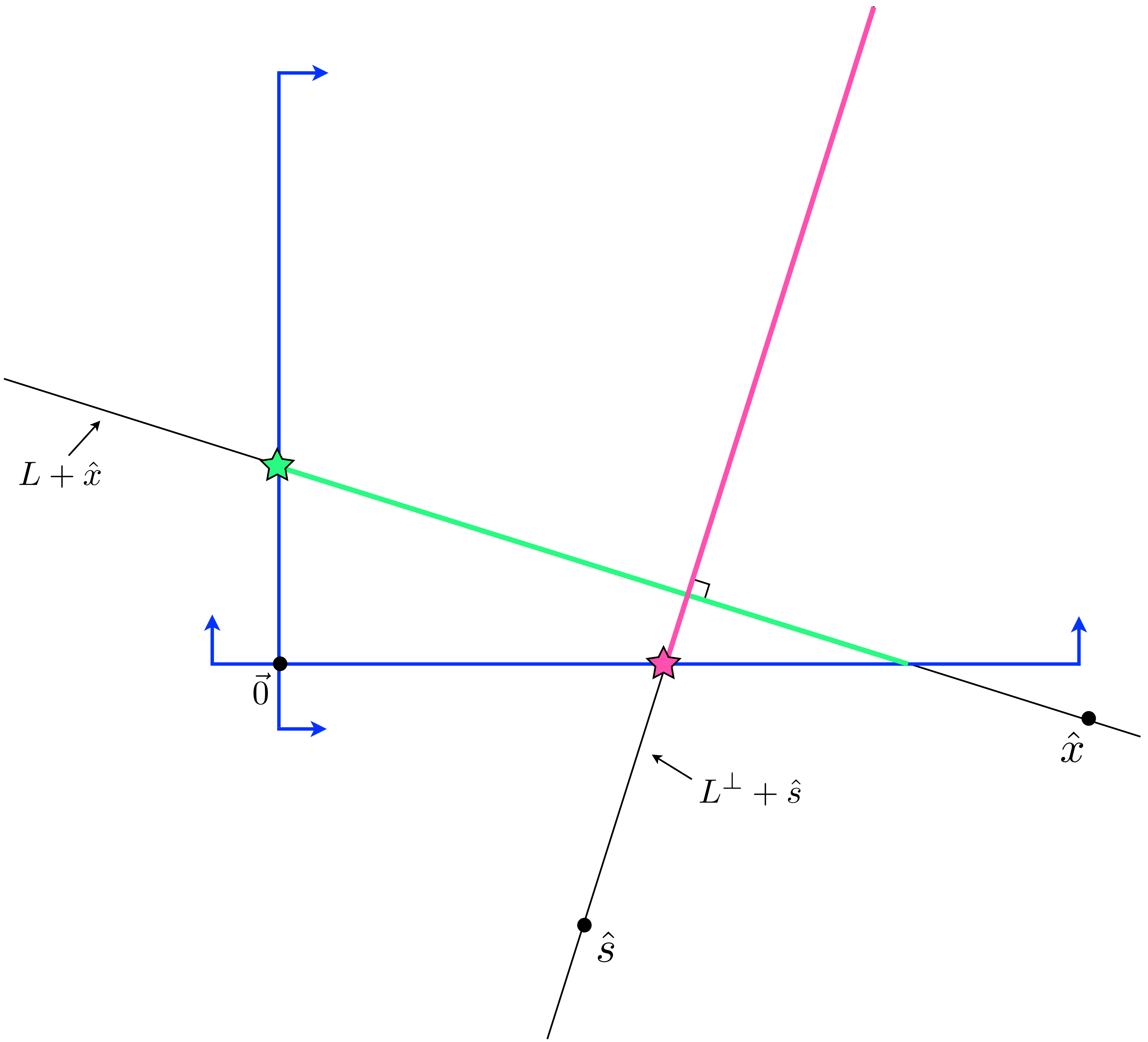
“min” rather than “max”

$$\begin{array}{rl} \boxed{\min} & c^T x \\ \text{s.t.} & x \in L + \hat{x} \\ & x \geq 0 \end{array}$$

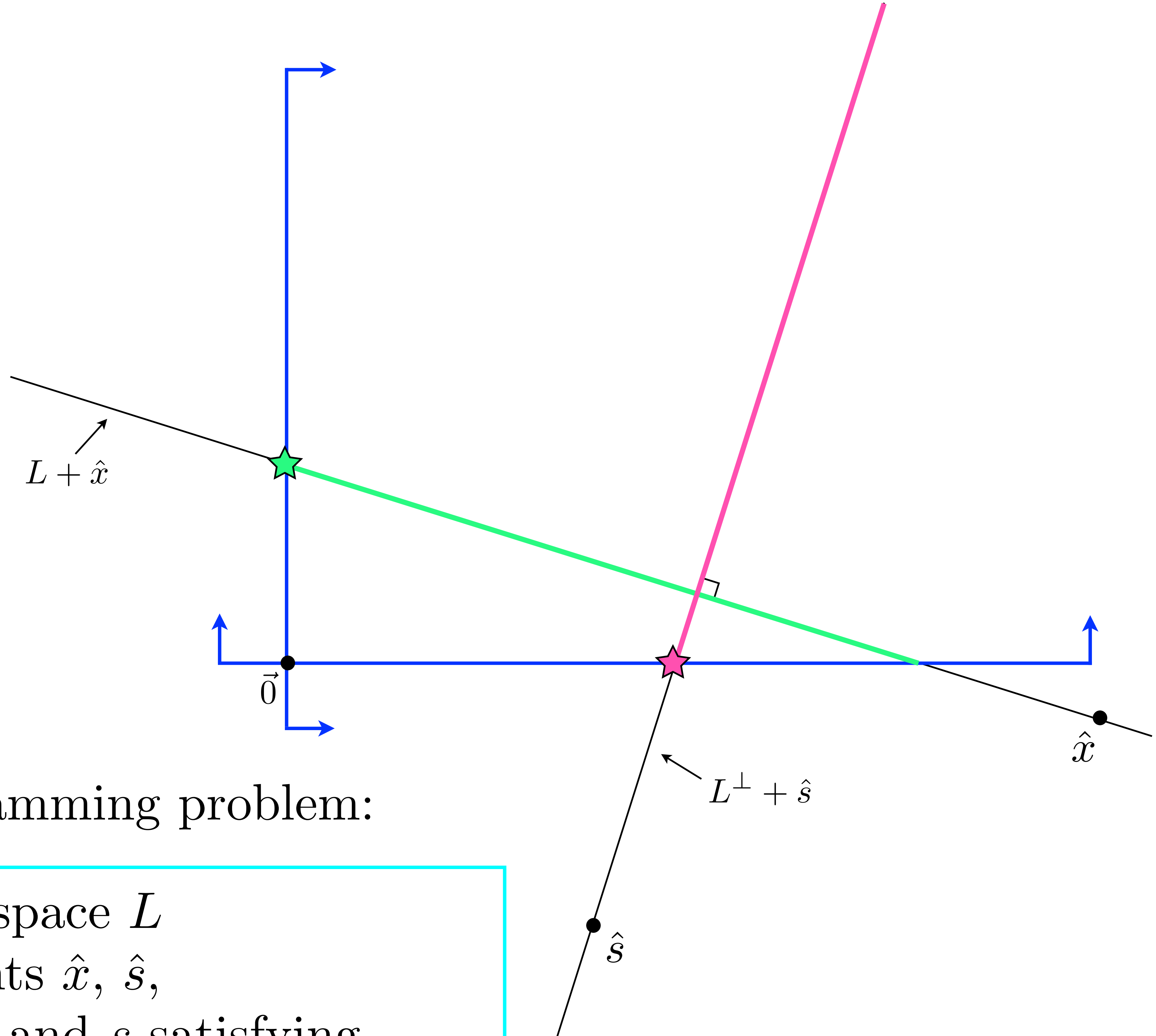


“dual linear program”

$$\begin{array}{rl} \min_s & \hat{x}^T s \\ \text{s.t.} & s \in L^{\perp} + c \\ & s \geq 0 \end{array}$$



$L + \hat{x}$
$\vec{0}$
$\hat{x}$
$L^{\perp} + \hat{s}$
$\hat{s}$



The primal-dual
linear programming problem:

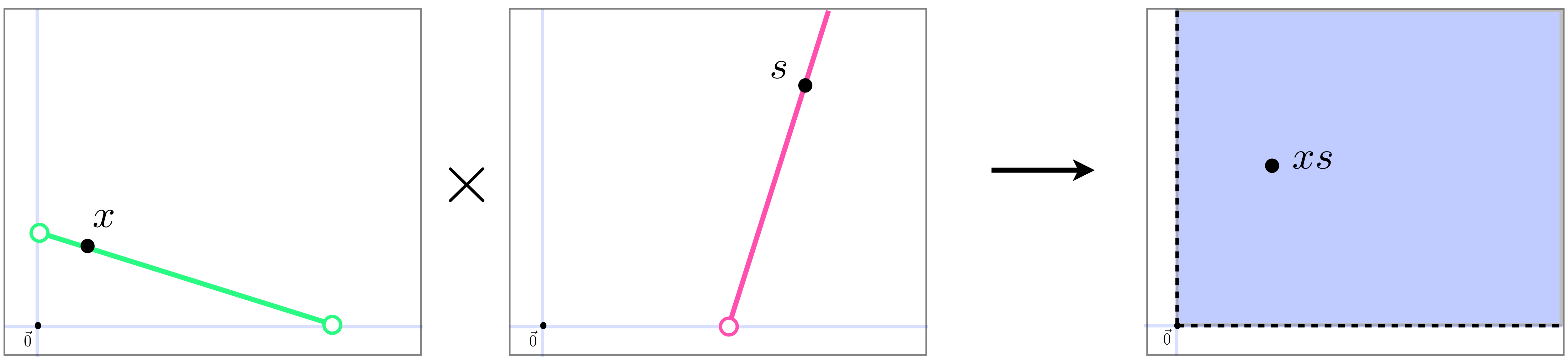
Given a subspace L
and points $\hat{x}$, $\hat{s}$,
find x and s satisfying

$$x \in L + \hat{x} \qquad x, s \geq 0 \qquad x^T s = 0$$
$$s \in L^{\perp} + \hat{s}$$

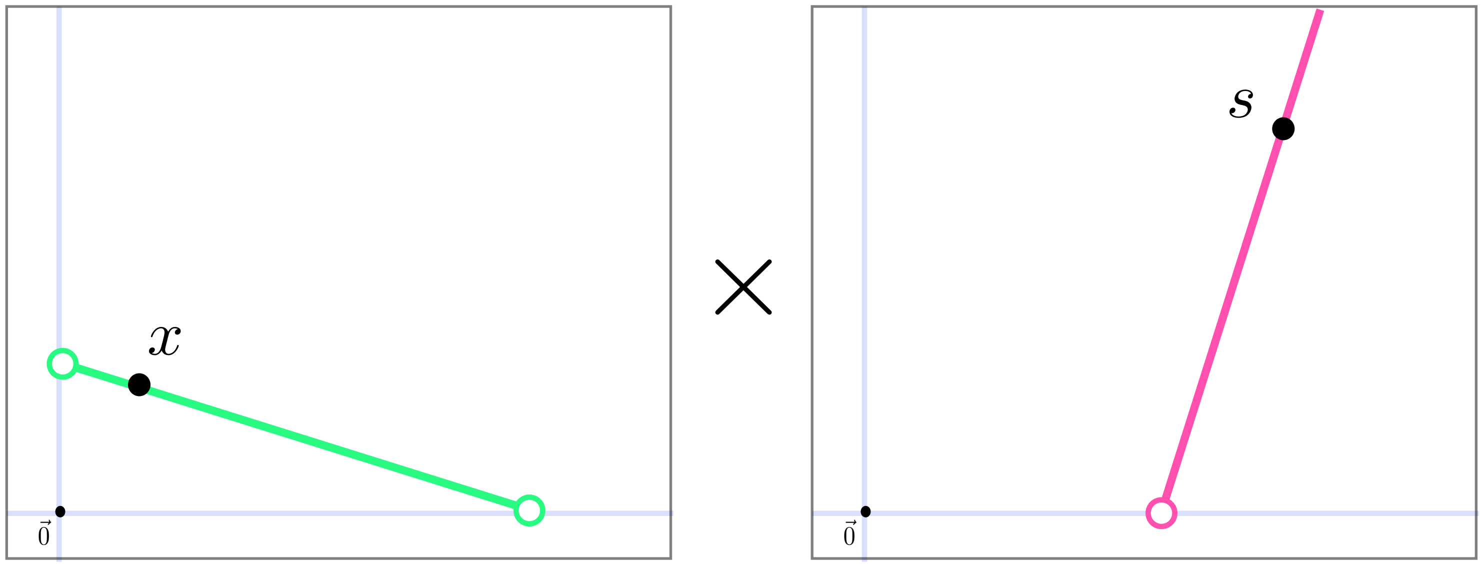
$$\mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}^n$$

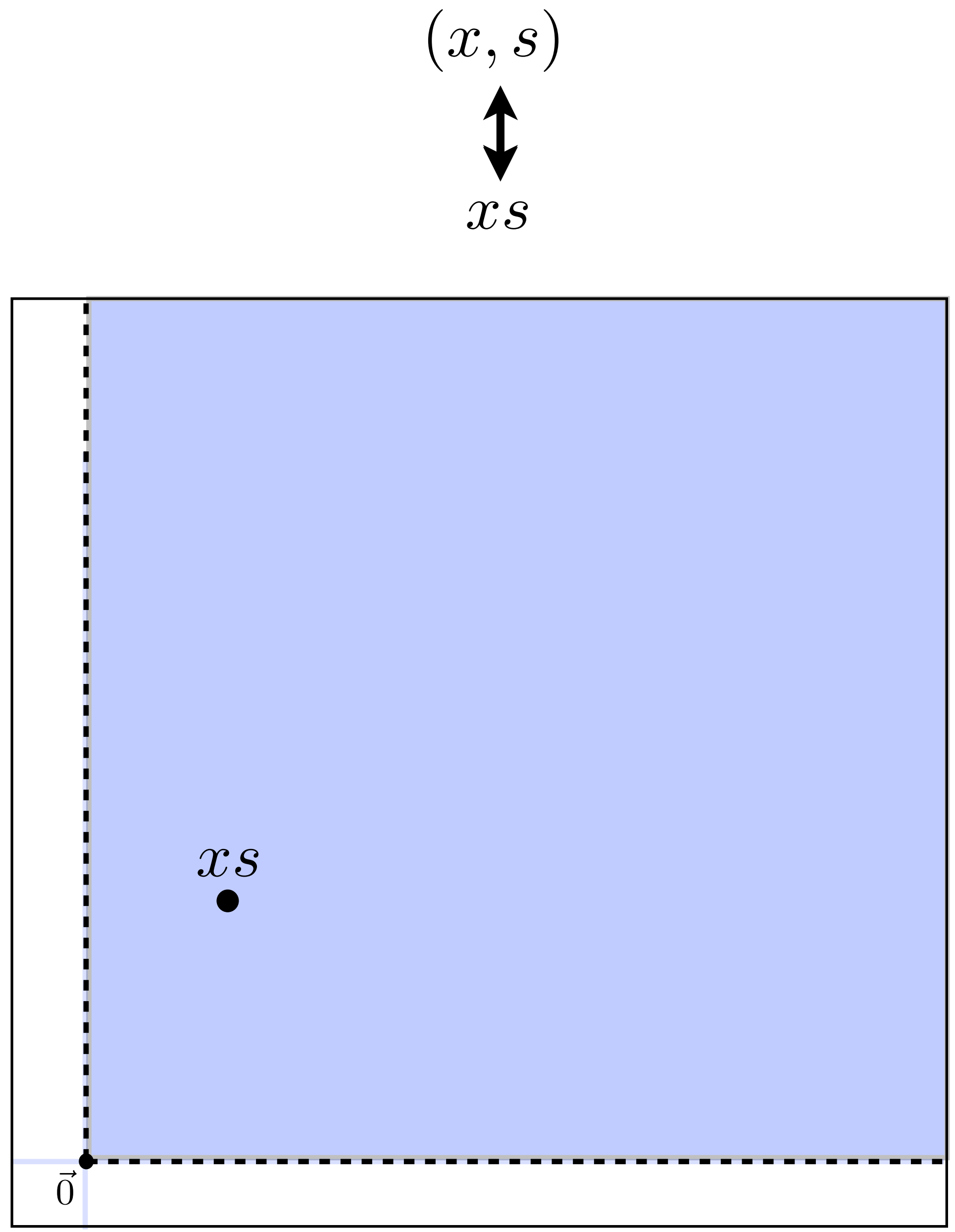
$$(x, s) \mapsto xs := \begin{bmatrix} x_1 s_1 \\ \vdots \\ x_n s_n \end{bmatrix}$$

Restricted map



is a diffeo with $\mathbb{R}^n_{++}$

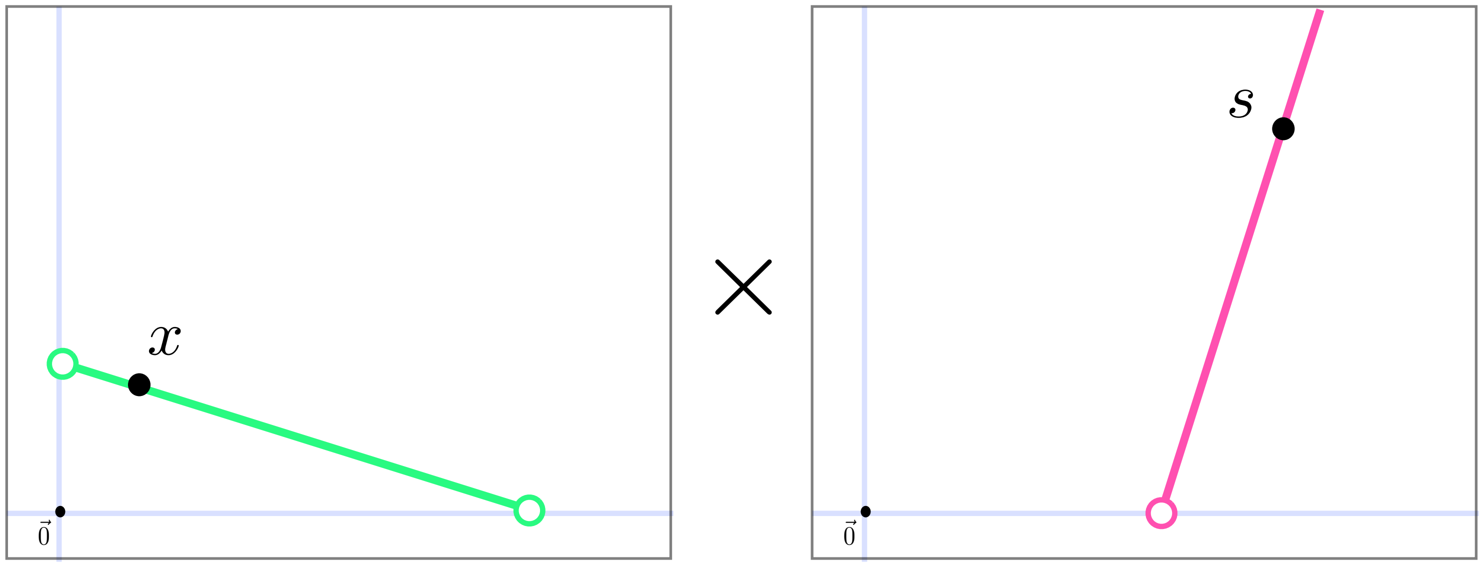




The Riemannian structure induced on $\mathbb{R}^n_{++}$
depends on the particular affine spaces
$L + \hat{x}$ and $L^\perp + \hat{s}$

For algorithmic purposes,
it is useful to endow $\mathbb{R}^n_{++}$ with a different
– albeit particularly elementary –
Riemannian structure

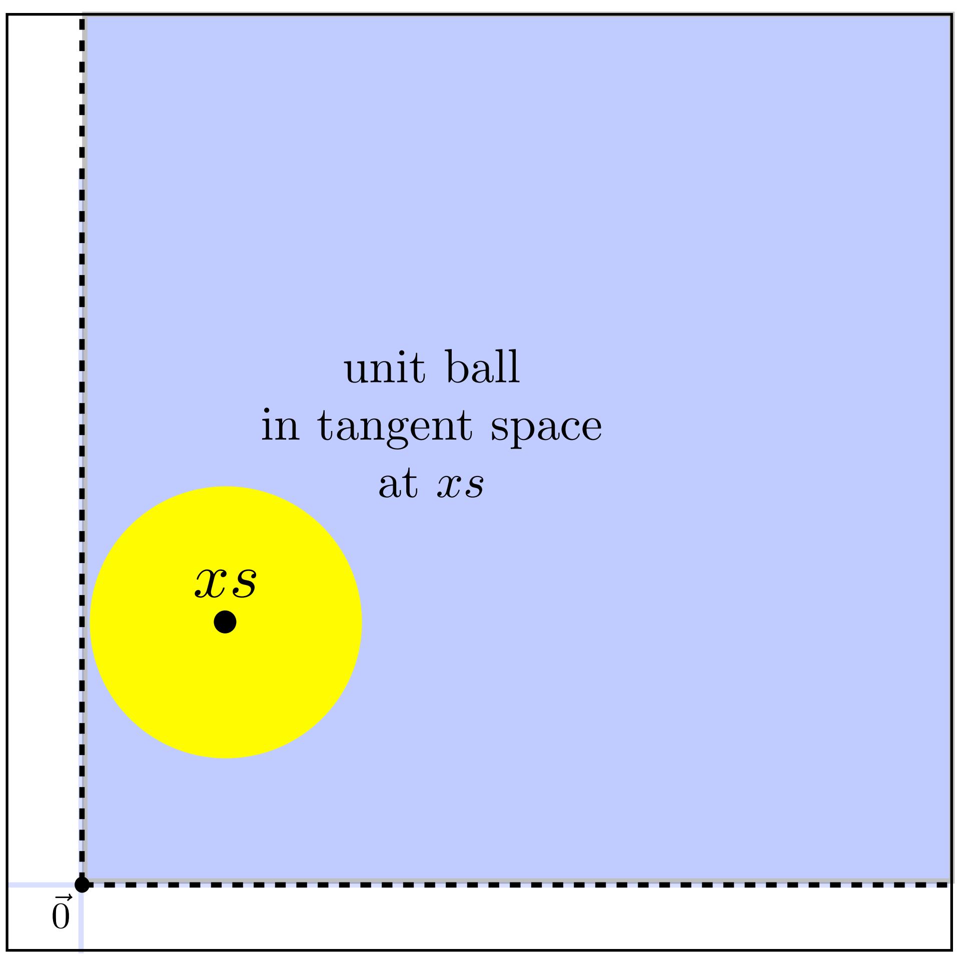
Specifically, for $v \in \mathbb{R}^n_{++}$,
let
$\langle u, \hat{u} \rangle_v := u^T \hat{u} / (\min v_j)^2$



(x, s)

$\updownarrow$

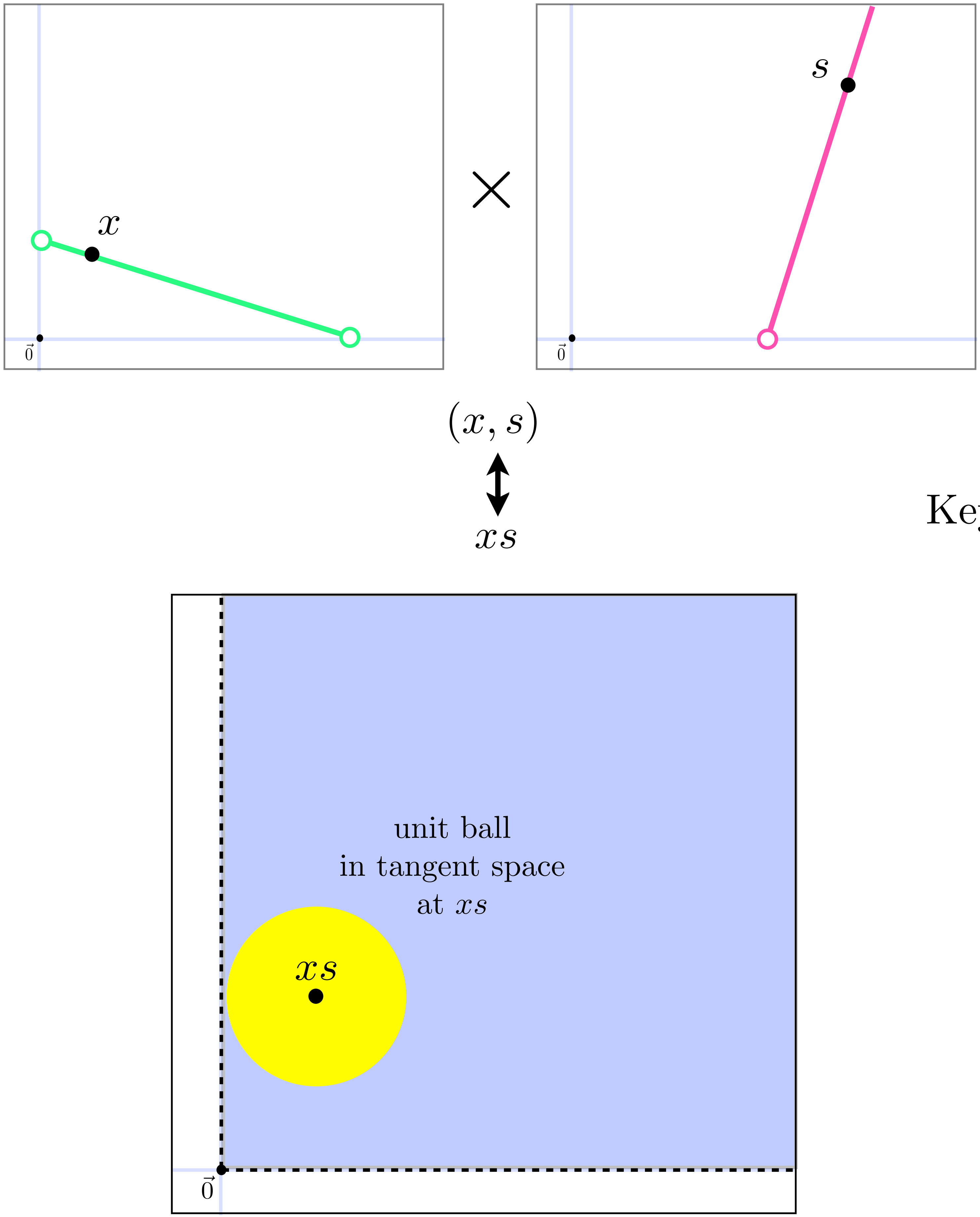
xs



The Riemannian structure induced on $\mathbb{R}^n_{++}$
depends on the particular affine spaces
$L + \hat{x}$ and $L^\perp + \hat{s}$

For algorithmic purposes,
it is useful to endow $\mathbb{R}^n_{++}$ with a different
– albeit particularly elementary –
Riemannian structure

Specifically, for $v \in \mathbb{R}^n_{++}$,
let
$\langle u, \hat{u} \rangle_v := u^T \hat{u} / (\min v_j)^2$

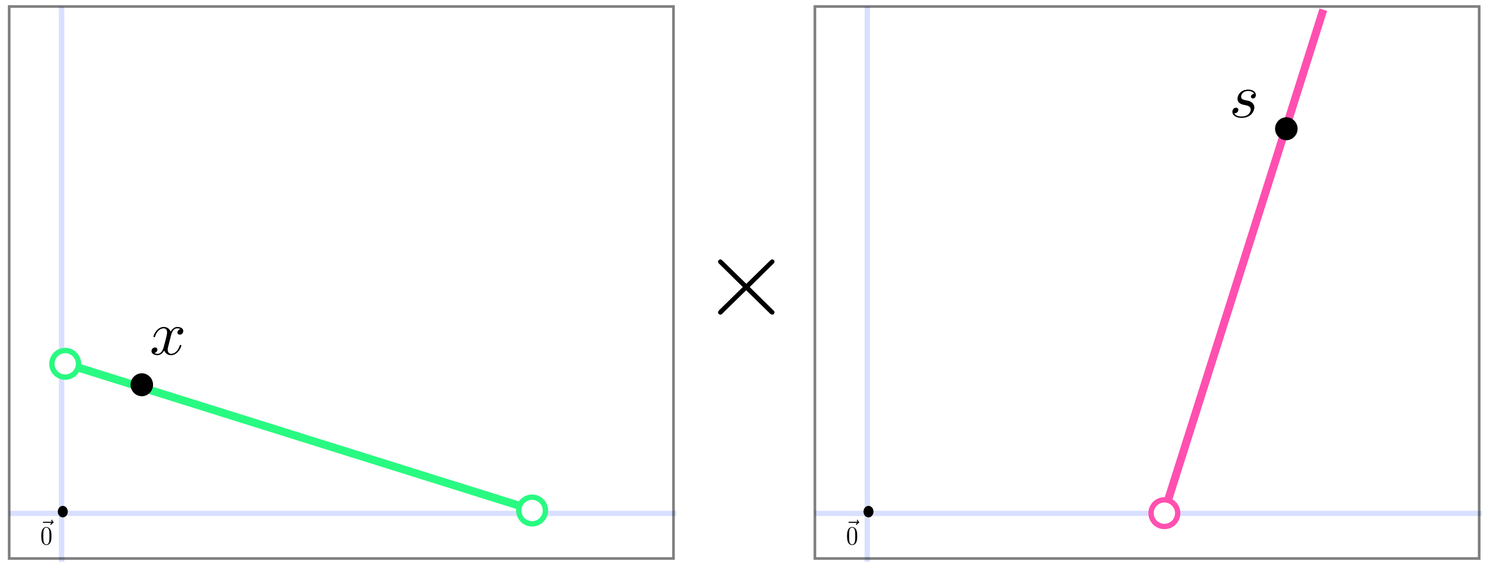


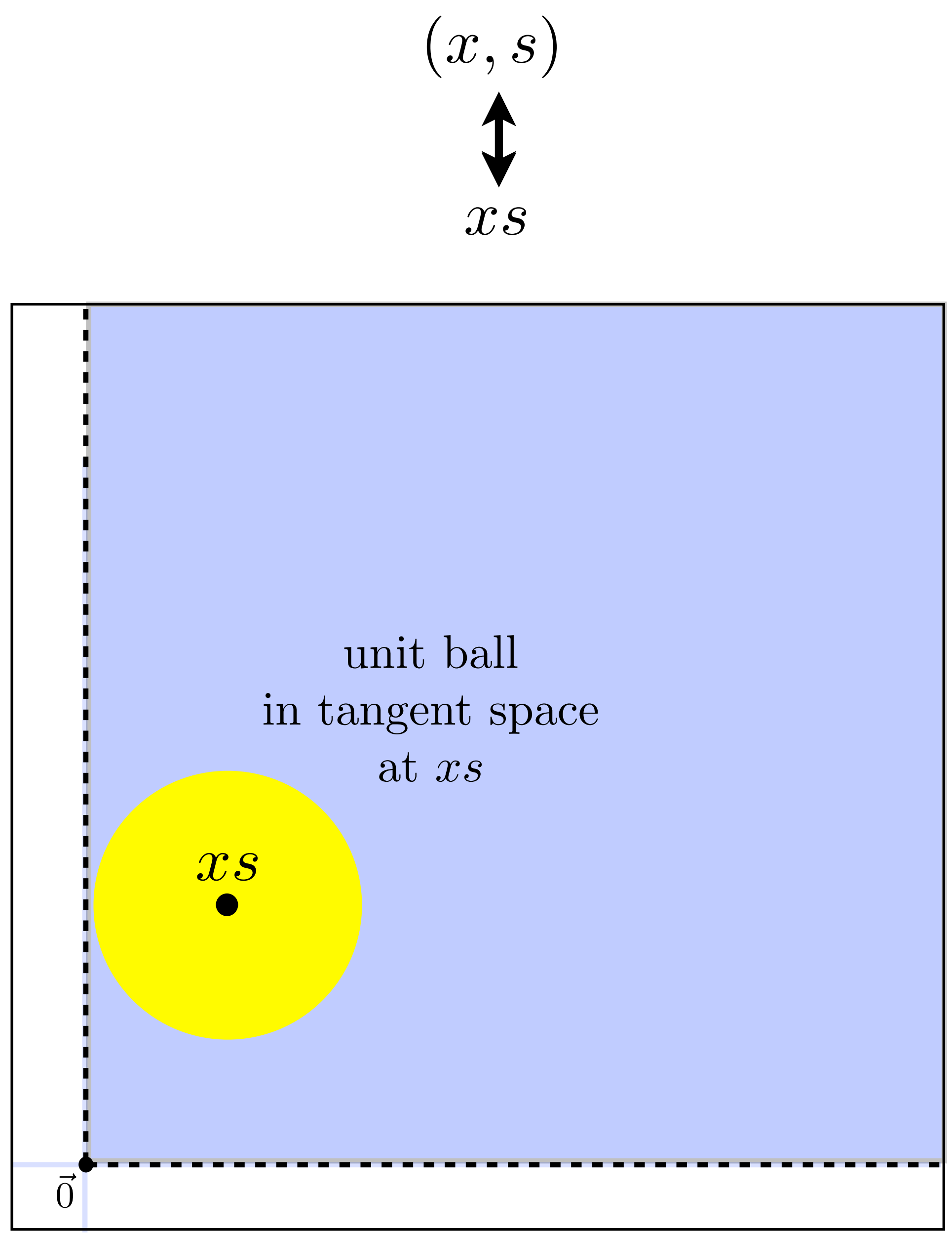
Key fact:

$$\text{The image of } \overbrace{B_{(x,s)}(0,1)}^{\substack{\text{unit ball}\\ \text{in tangent space}\\ \text{at } (x,s)}}$$

$$\text{under the differential for } \downarrow$$

$$\text{covers } \underbrace{B_{xs}(0,1)}_{\substack{\text{unit ball}\\ \text{in tangent space}\\ \text{at } xs}}$$



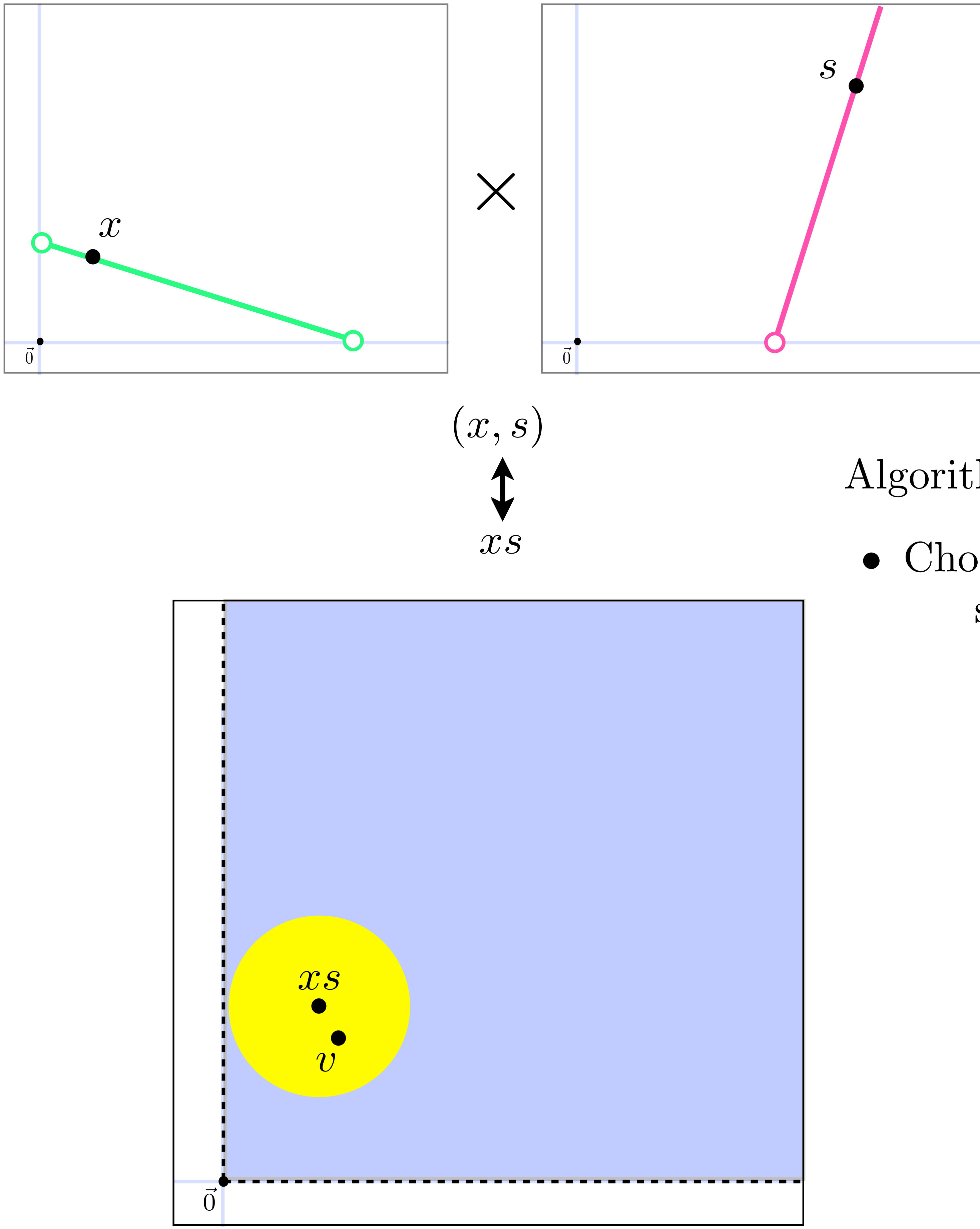


Key fact:

The image of $B_{(x,s)}(0,1)$
under the differential for $\downarrow$
covers $B_{xs}(0,1)$

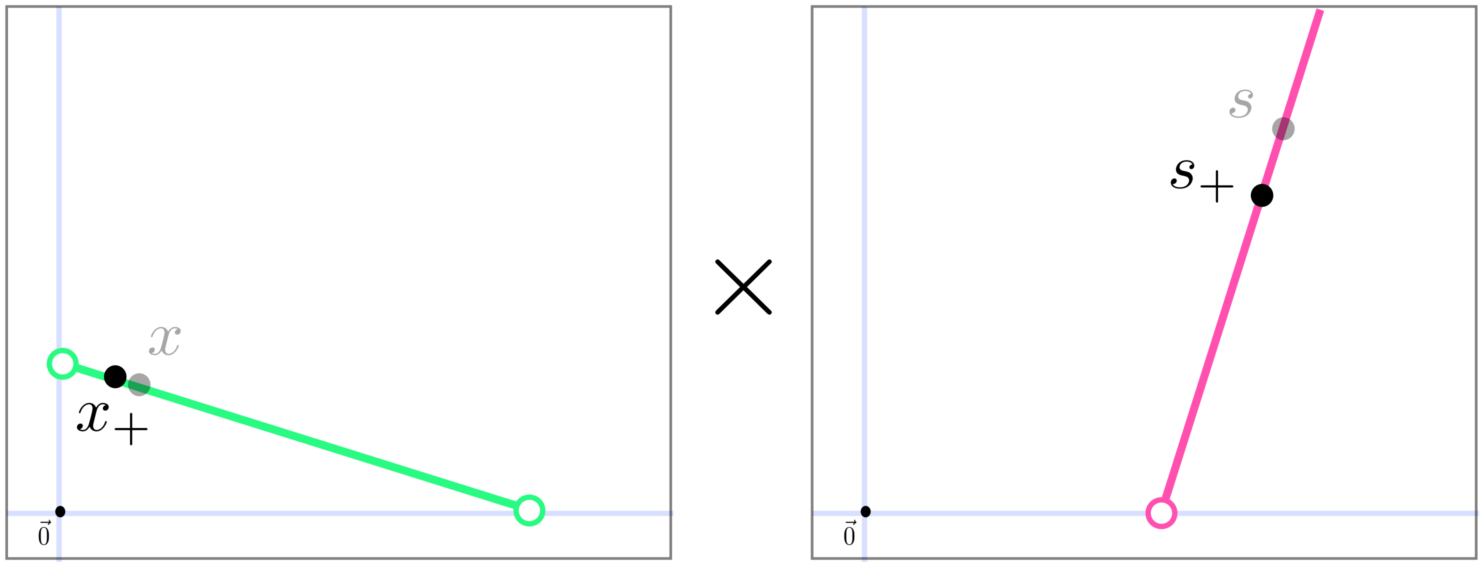
Consequence:

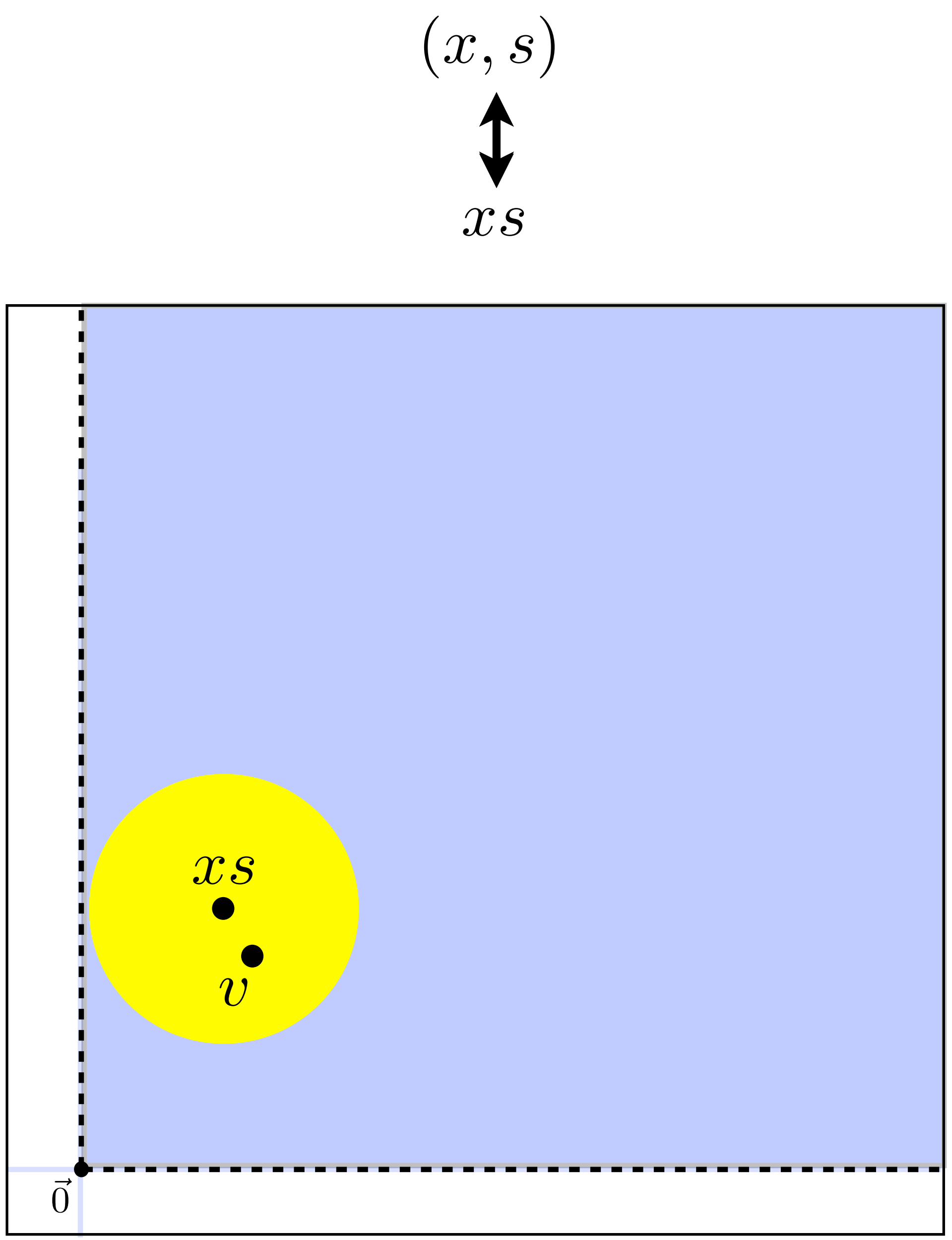
The image of $u \in B_{xs}(0,1)$
under the differential for $\uparrow$
is a vector $(\Delta x, \Delta s)$
for which $x + \Delta x$ and $s + \Delta s$
are feasible.



Algorithm:

- Choose "target" v
  satisfying $\|v - xs\|_{xs} < 1$

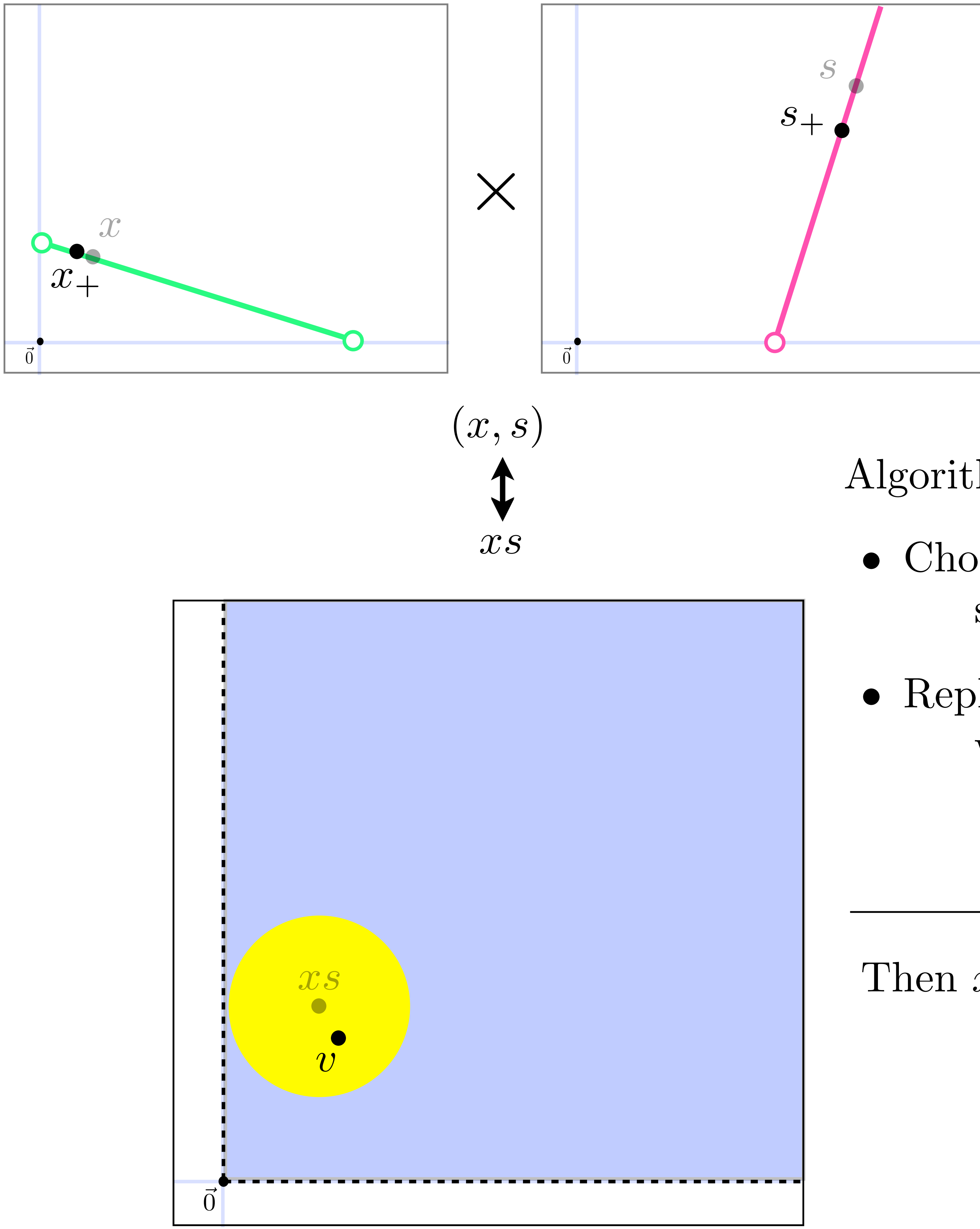




Algorithm:

- Choose "target" v
  satisfying $\|v - xs\|_{xs} < 1$

- Replace map $(x, s) \mapsto xs$
  with first-order approximation at (x, s)
  and let (x_+, s_+) be the pair
  mapping to v

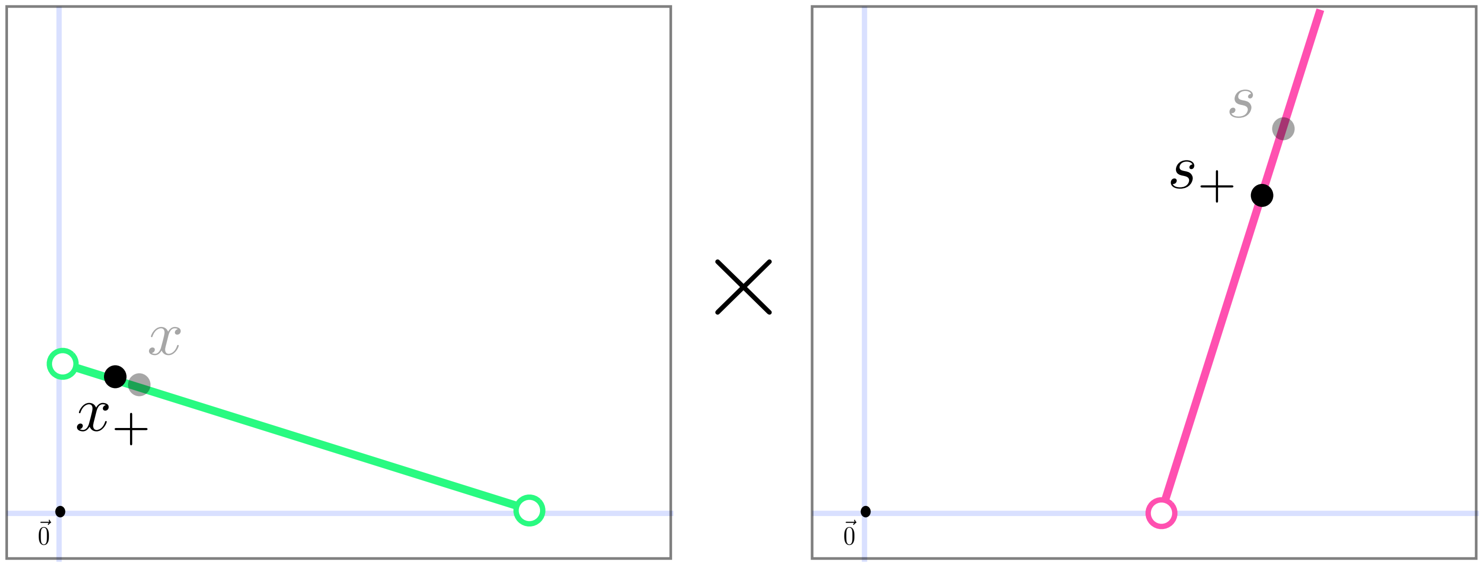


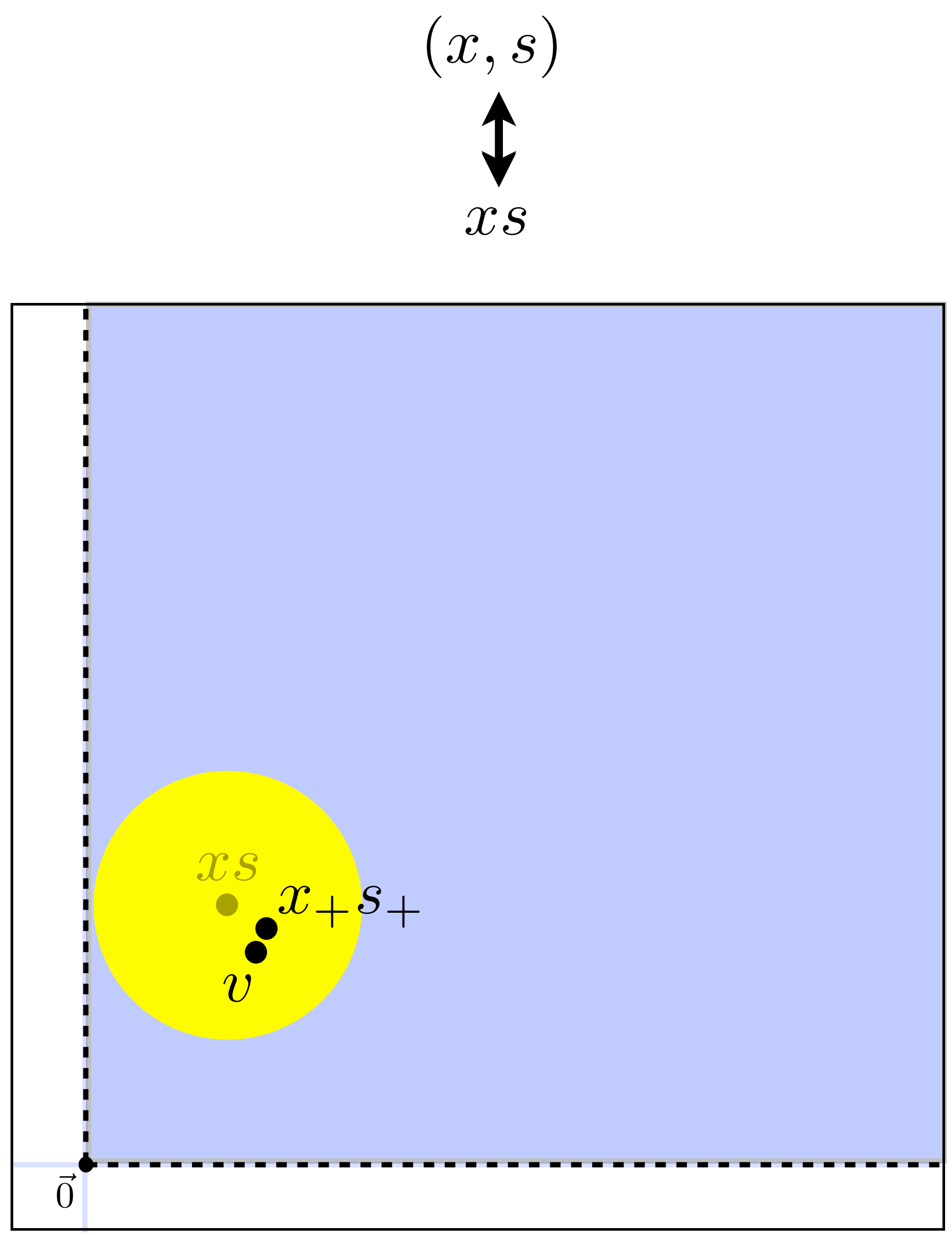
Algorithm:

- Choose "target" v
  satisfying $\|v - xs\|_{xs} < 1$

- Replace map $(x, s) \mapsto xs$
  with first-order approximation at (x, s)
  and let (x_+, s_+) be the pair
  mapping to v

Then x_+ and s_+ are feasible and . . .





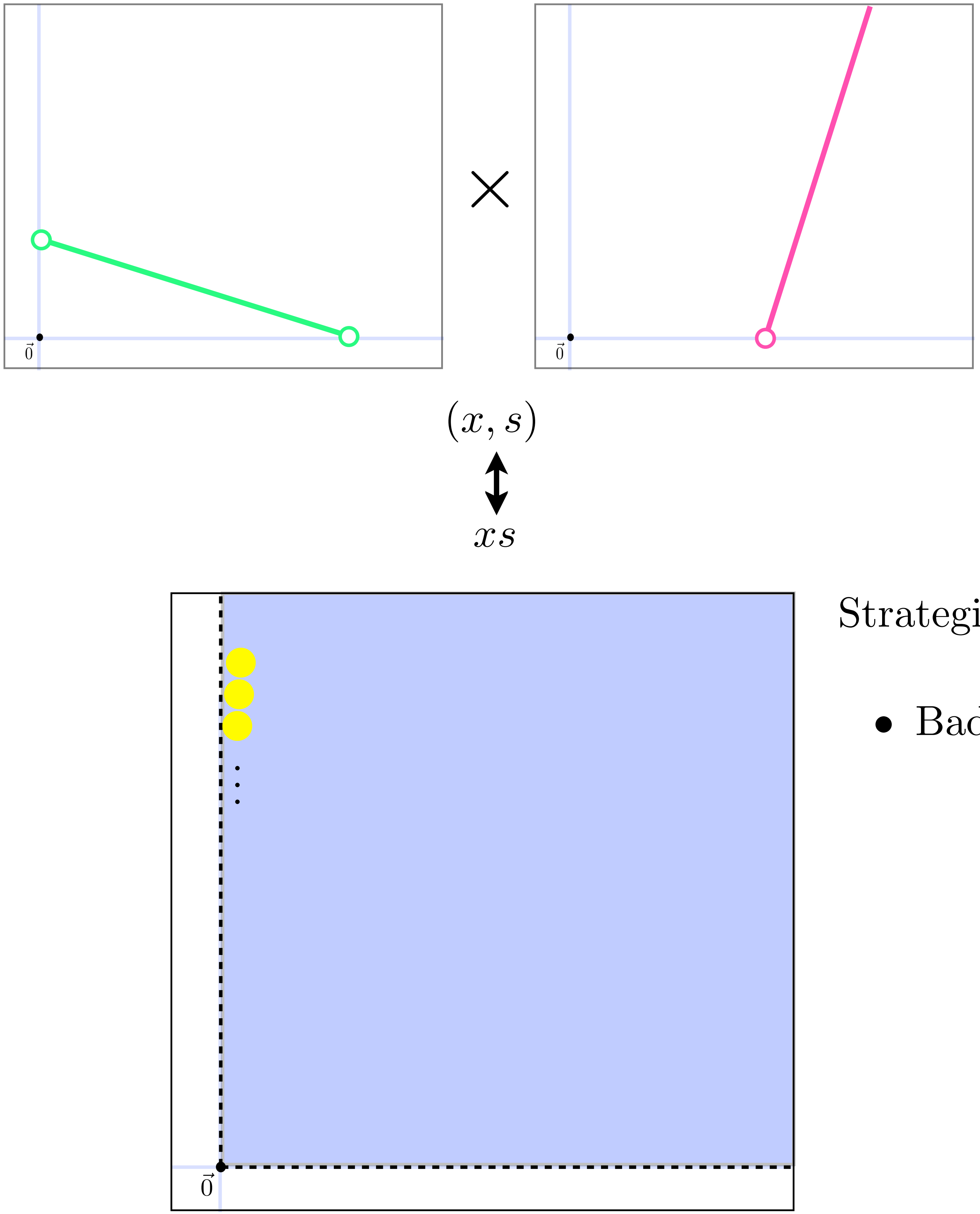
Algorithm:

- Choose "target" v
  satisfying $\|v - xs\|_{xs} < 1$
- Replace map $(x, s) \mapsto xs$
  with first-order approximation at (x, s)
  and let (x_+, s_+) be the pair
  mapping to v

Then x_+ and s_+ are feasible and ...

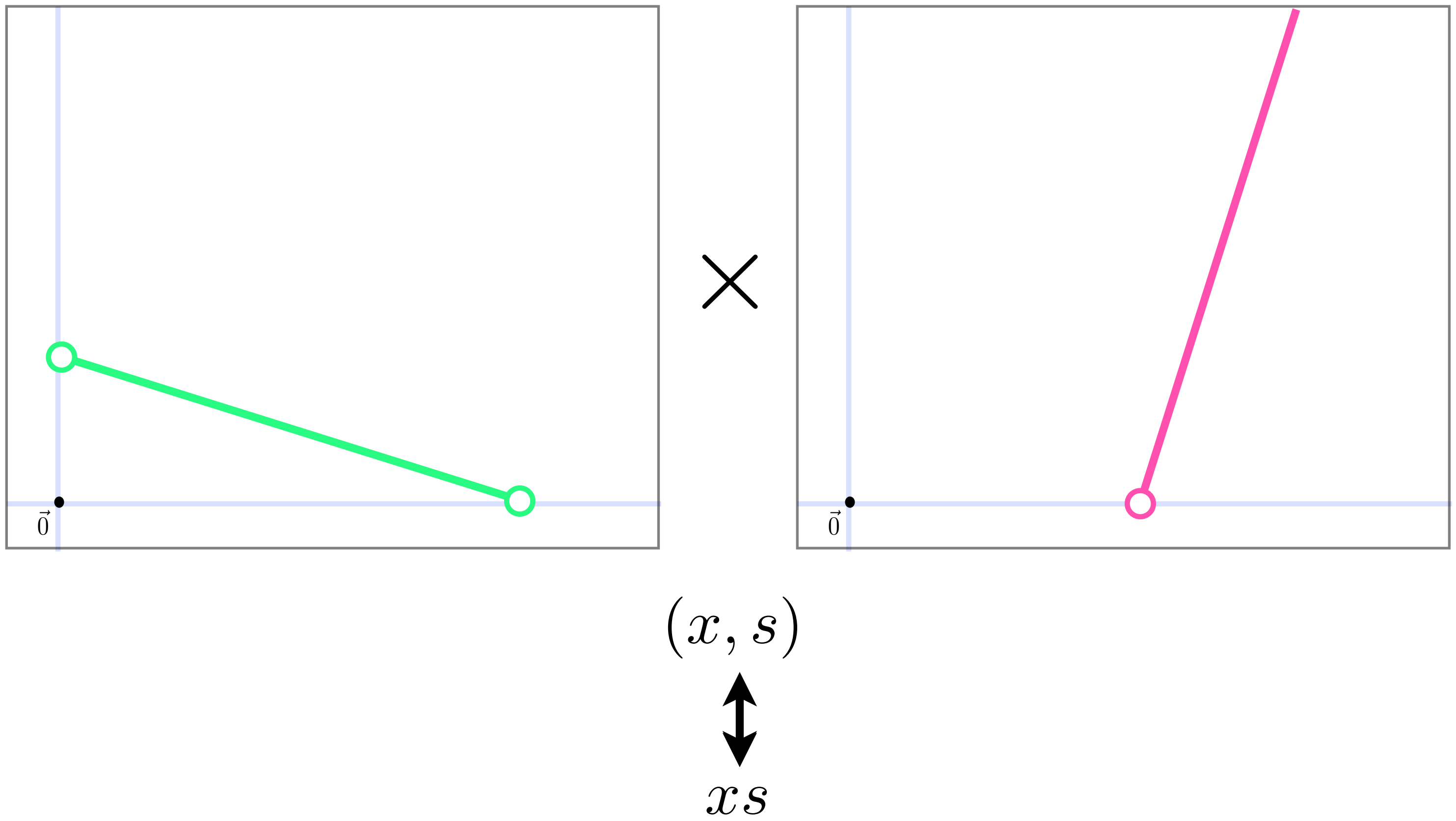
Fact:

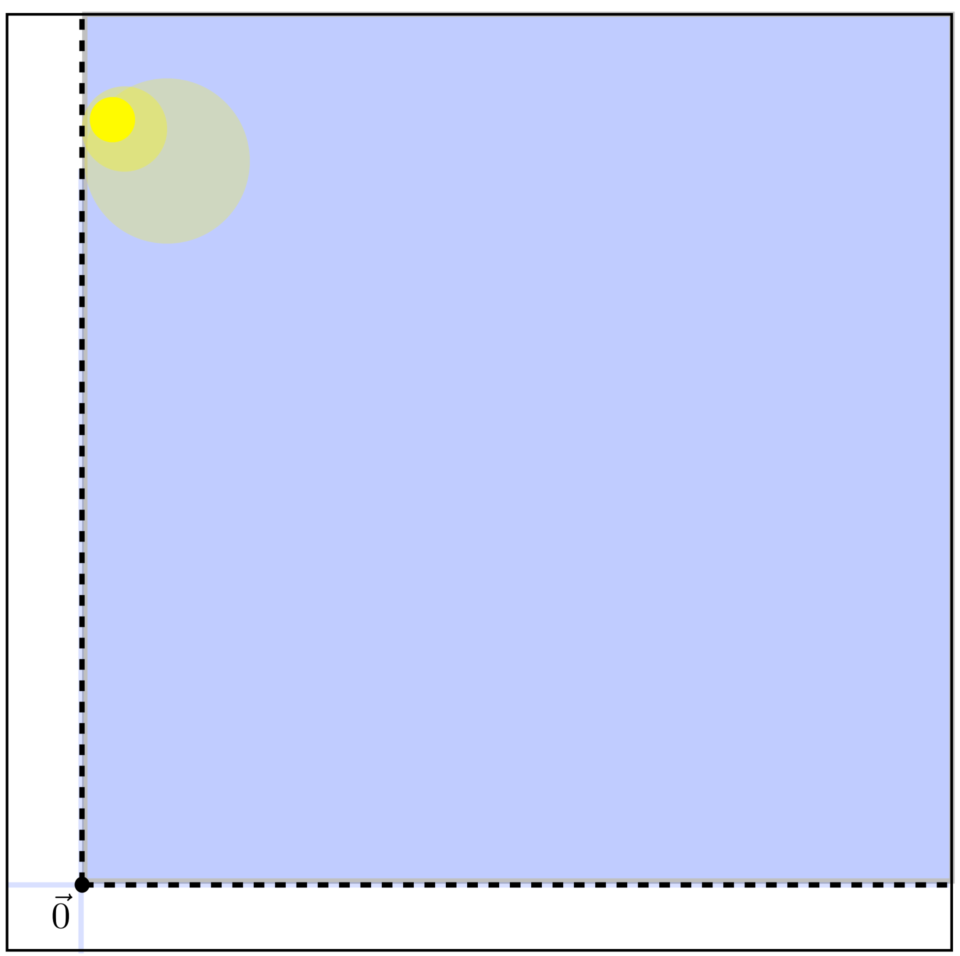
$$\|x_+s_+ - v\|_{xs} \leq \frac{1}{2}\|v - xs\|_{xs}^2$$



Strategies in choosing the target v:

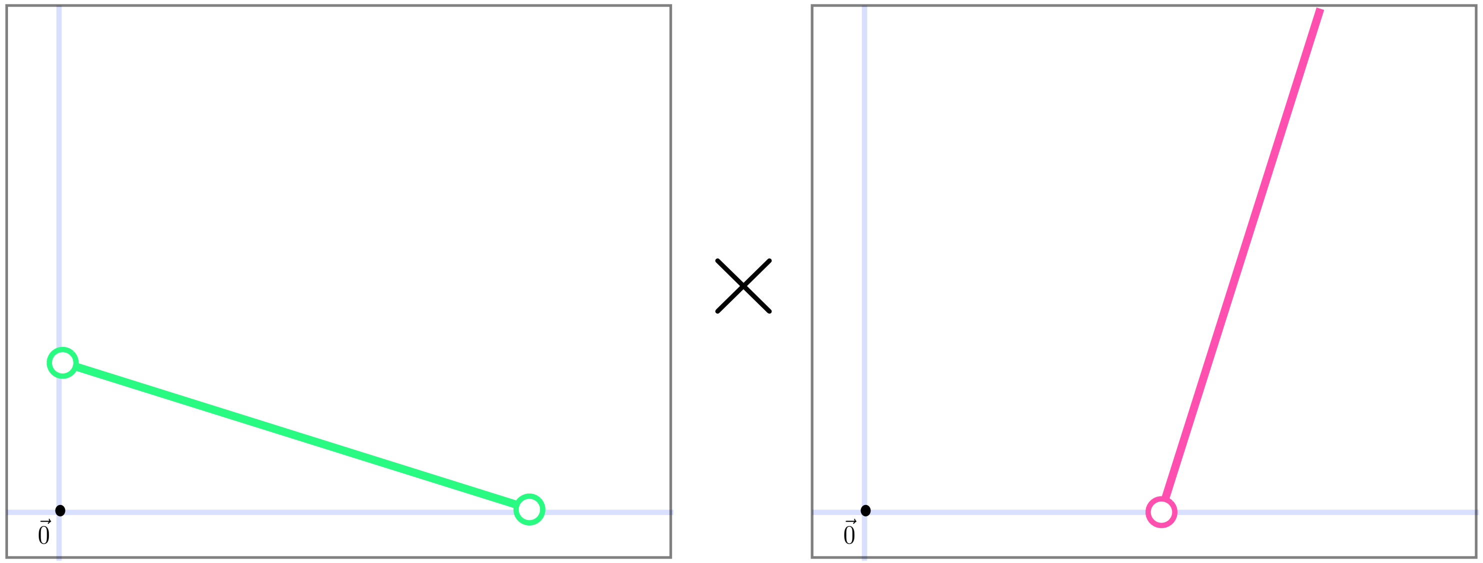
- Bad: Move directly towards the origin.





Strategies in choosing the target v:

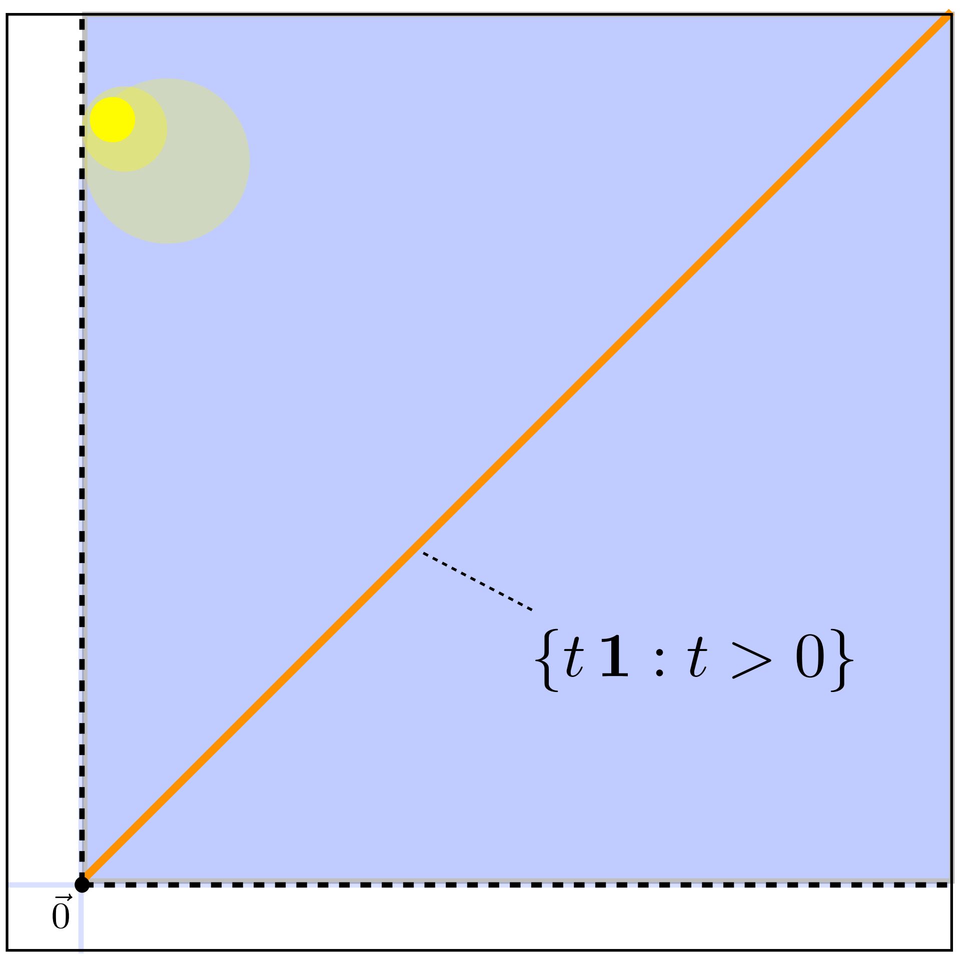
- Bad: Move directly towards the origin.
- Good: Get away from boundary,
  then move towards origin.



(x, s)

$\updownarrow$

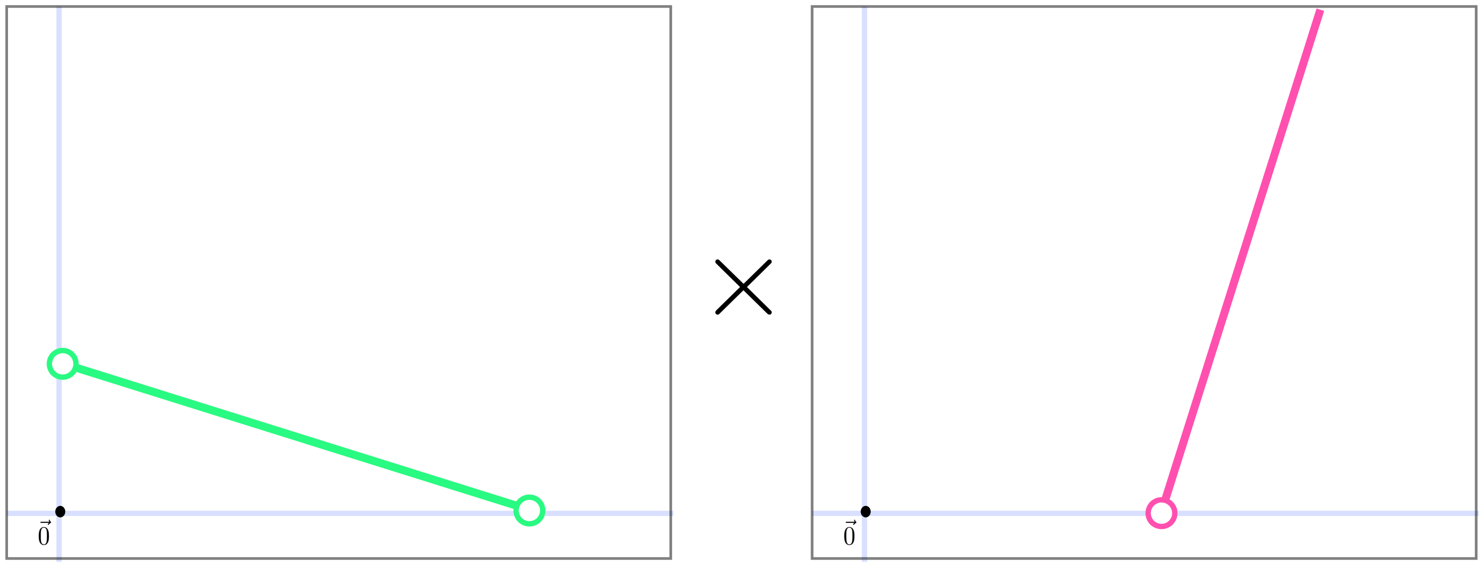
xs



Strategies in choosing the target v:

- Bad: Move directly towards the origin.
- Good: Get away from boundary,
  then move towards origin.

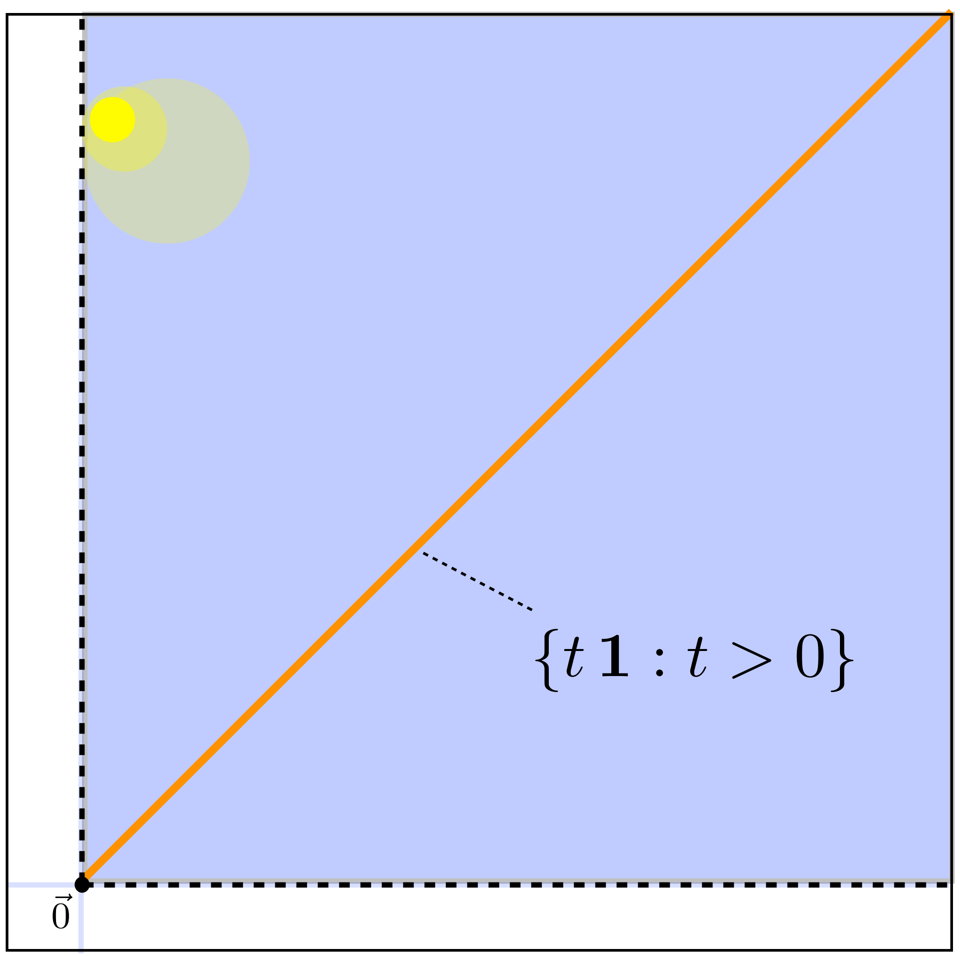
  that is, start by moving towards
  the line $\{t\mathbf{1} : t > 0\}$



(x, s)

$\updownarrow$

xs



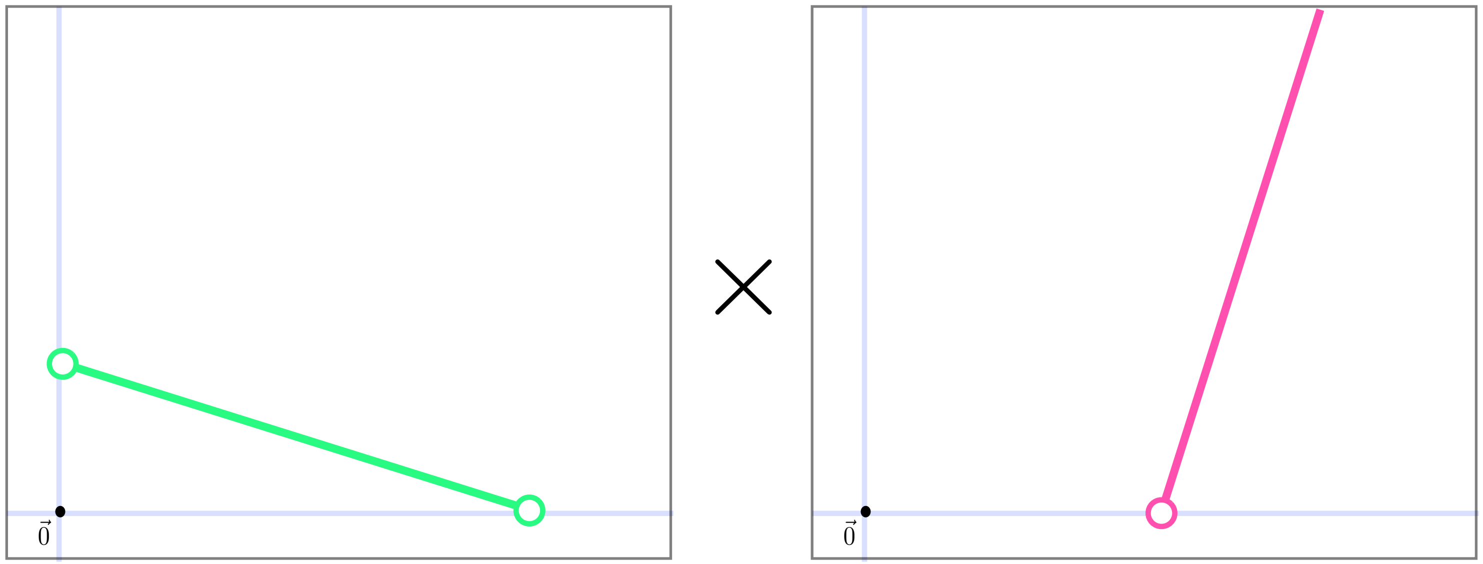
Strategies in choosing the target v:

- Bad: Move directly towards the origin.

- Good: Get away from boundary,
  then move towards origin.

  that is, start by moving towards
  the line $\{t\mathbf{1} : t > 0\}$

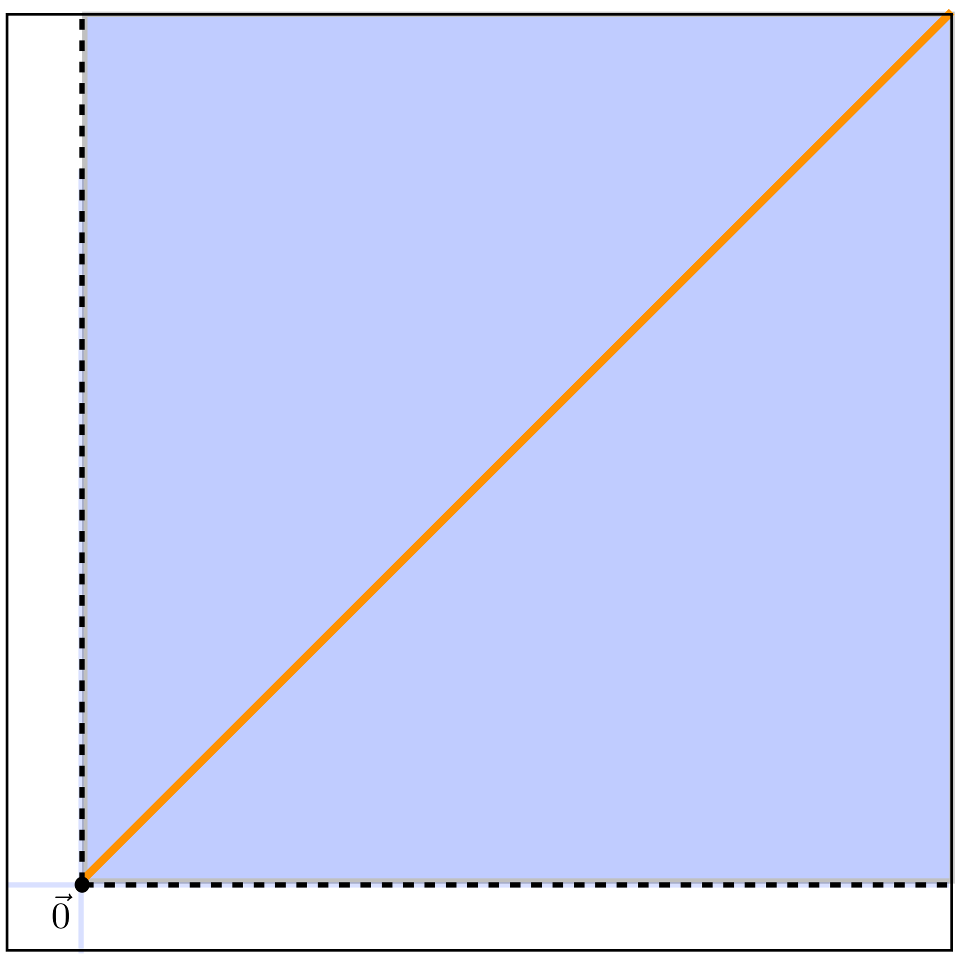
This line happens to be
the image of the primal-dual central path
under the map $(x, s) \mapsto xs$



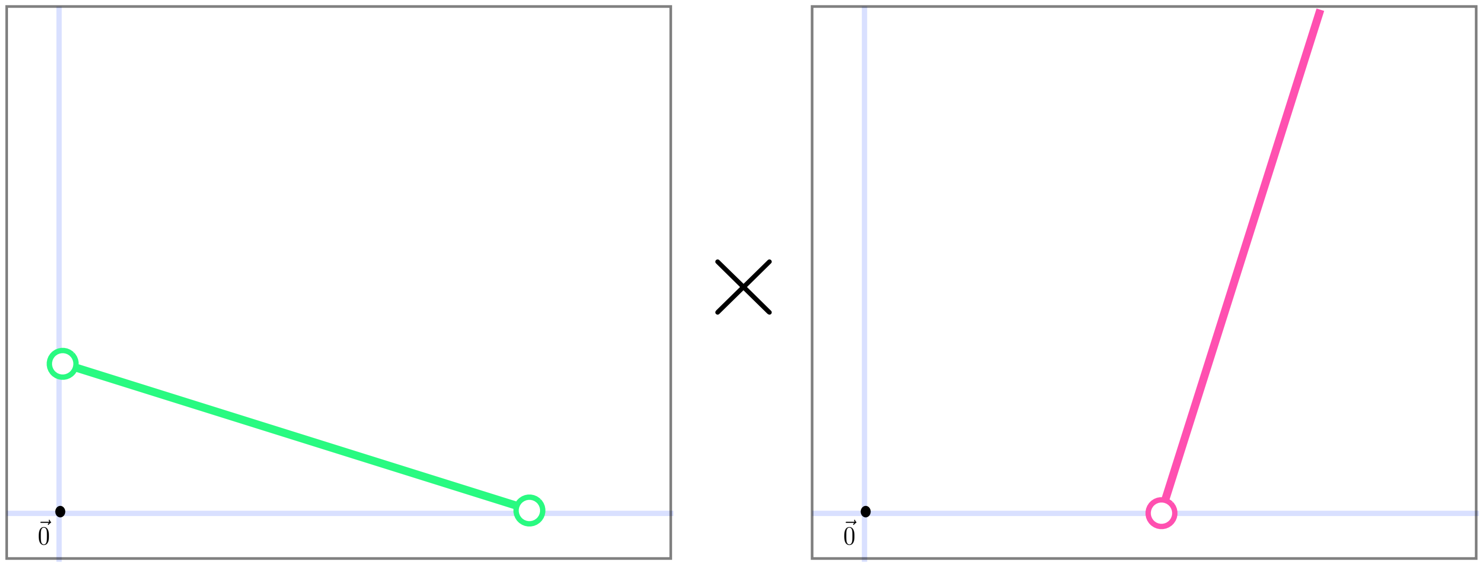
(x, s)

$\updownarrow$

xs



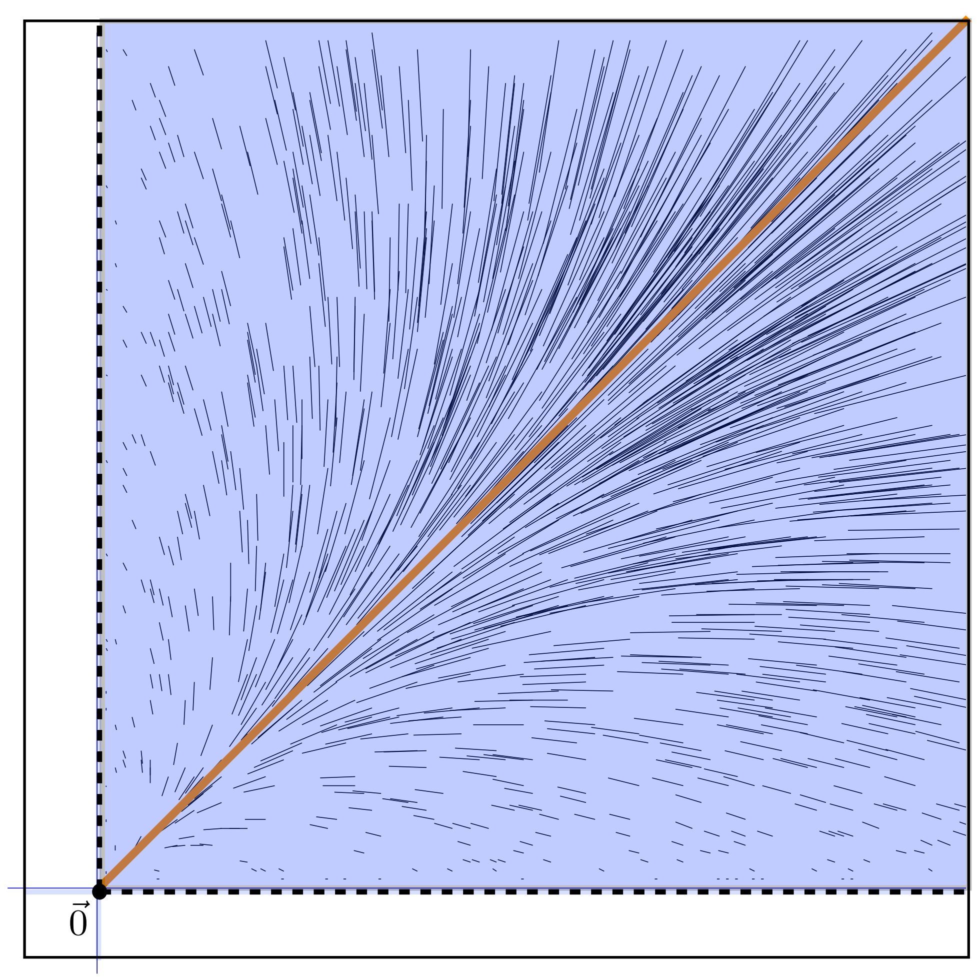
In designing primal-dual algorithms, vector field flows are thus most naturally defined on $\mathbb{R}^n_{++}$, then pulled back to the feasible regions.



(x, s)

$\updownarrow$

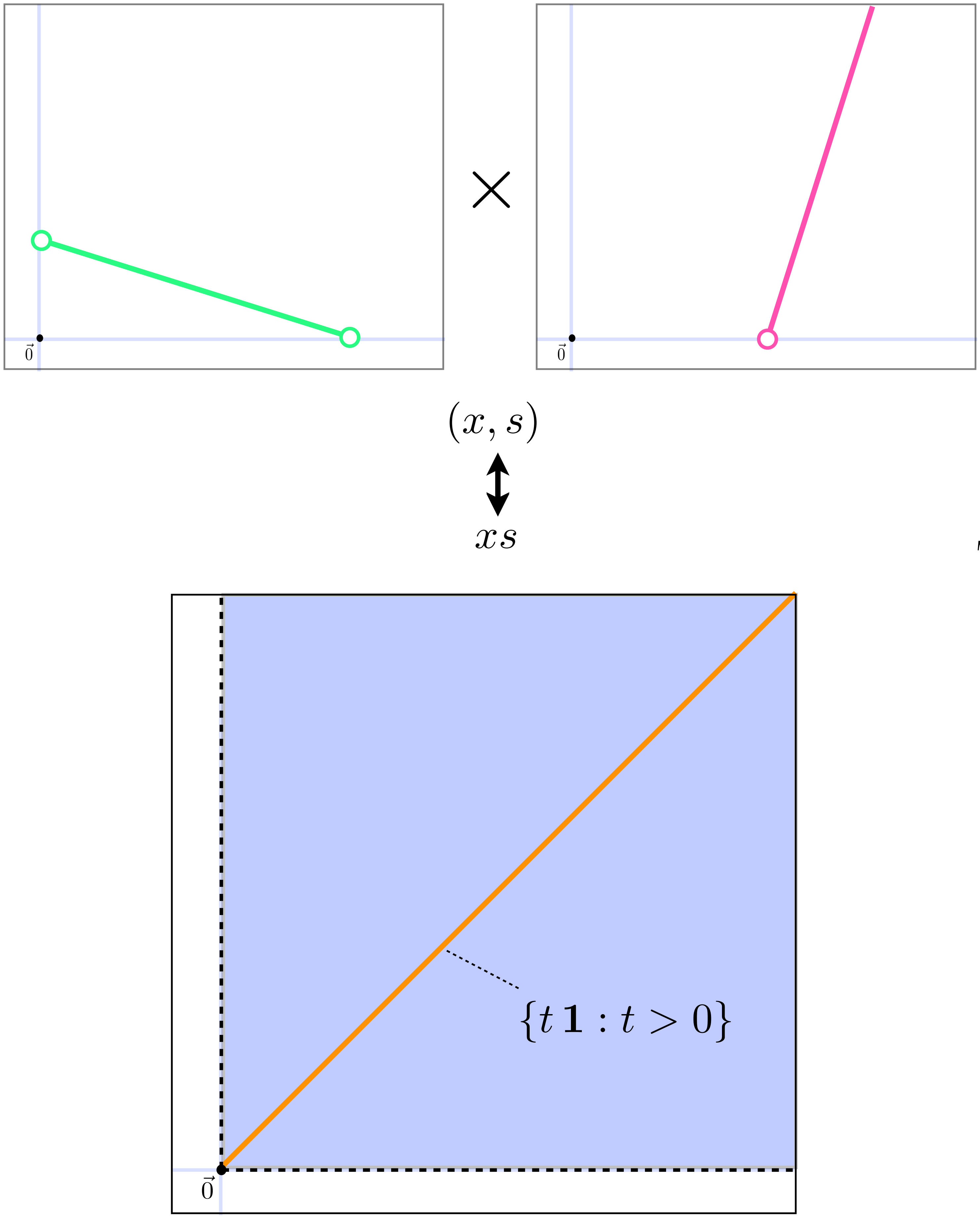
xs



In designing primal-dual algorithms,
vector field flows are thus
most naturally defined on $\mathbb{R}^n_{++}$,
then pulled back to the feasible regions.

For example,
the Tanabe-Todd-Ye
potential-reduction method
relies on the vector field $v \mapsto -v + \frac{\sum v_j}{n+\sqrt{n}}\mathbf{1}$

. . . and on the potential function
$(x, s) \mapsto (n + \sqrt{n}) \ln x^T s - \sum \ln x_j - \sum \ln s_j$



The primal-dual central path
is not necessarily a geodesic
but it is a "$\sqrt{2}$-geodesic"

– Nesterov & Todd (2002)

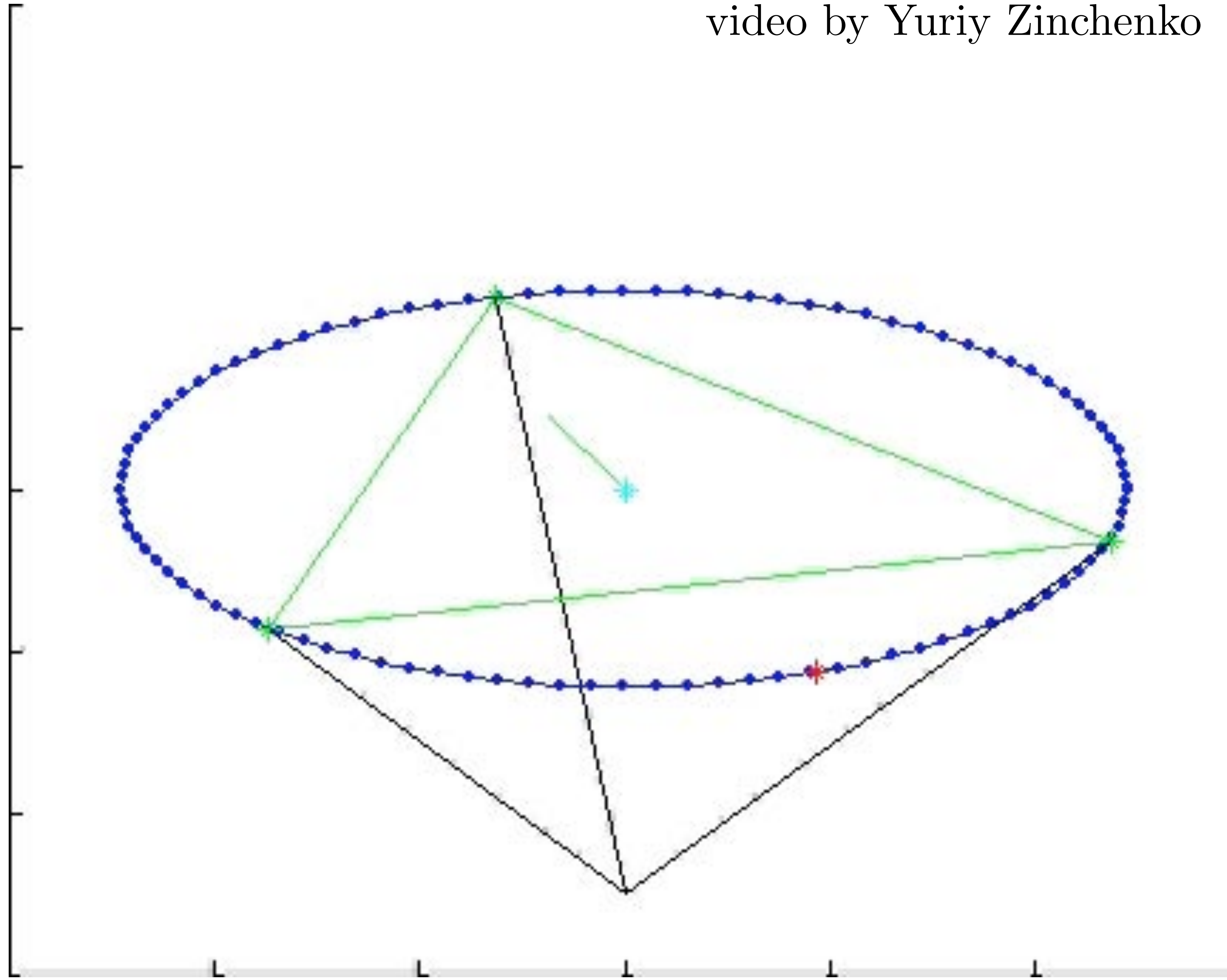
(results for
symmetric-cone programming,
not just for LP)

The primal (or dual) central path
is an $O(n^{1/4})$-geodesic

– Nesterov & Nemirovski (2008)

(results are *very* general)

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0 \end{array}$$

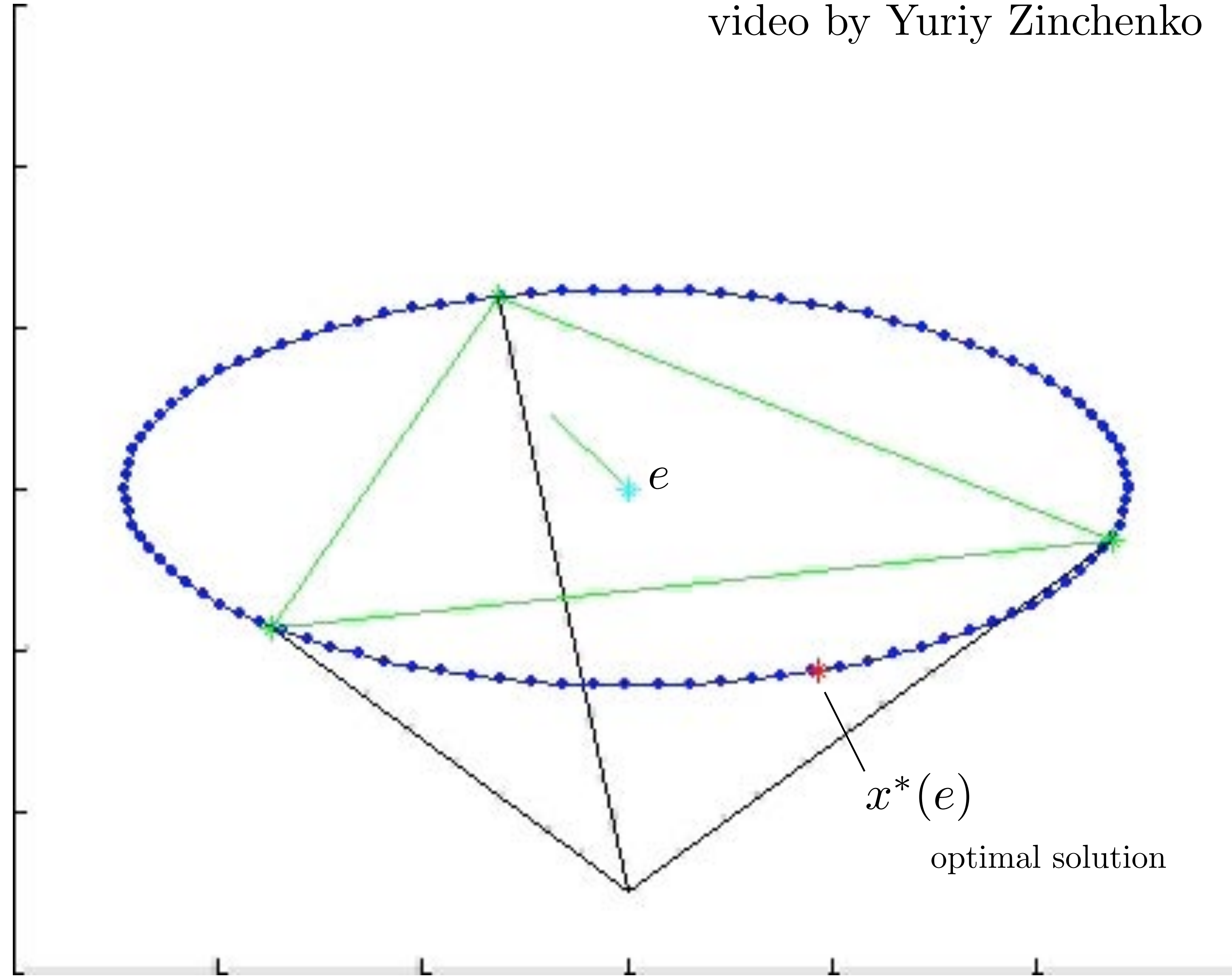


$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0 \end{array}$$

relaxation of LP:

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b \\ & x \in K(e) \end{array}$$

$$\{x : \textstyle\sum_{i<j} \frac{x_i x_j}{e_i e_j} \geq 0 \quad \text{and} \quad \sum_j \frac{x_j}{e_j} \geq 0\}$$



$$\begin{array}{ll} \max & c^T x \\ \text{s.t.} & Ax \geq b \end{array}$$

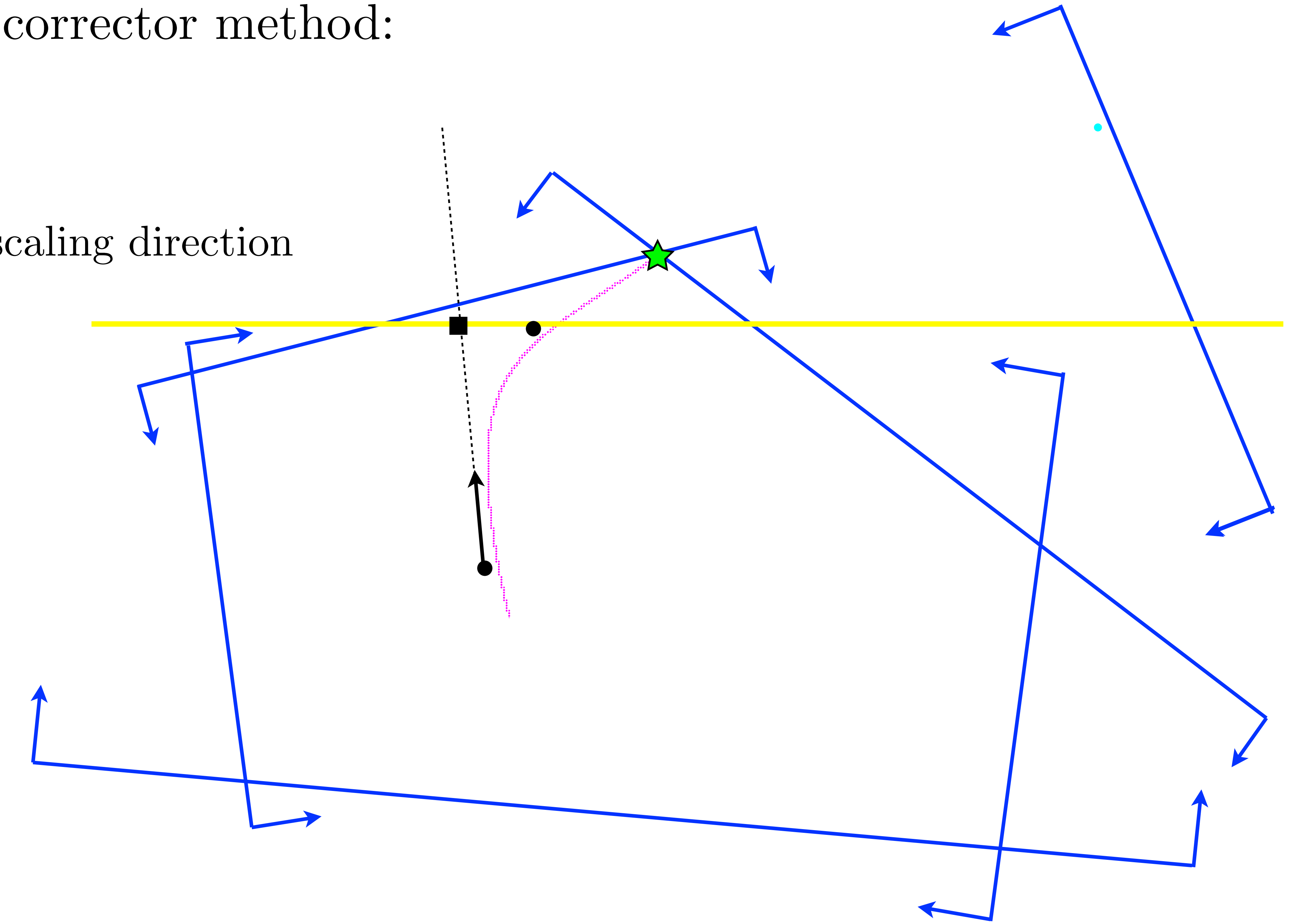
Recall the predictor-corrector method:

1) "Predict"

– move in affine-scaling direction

2) "Correct"

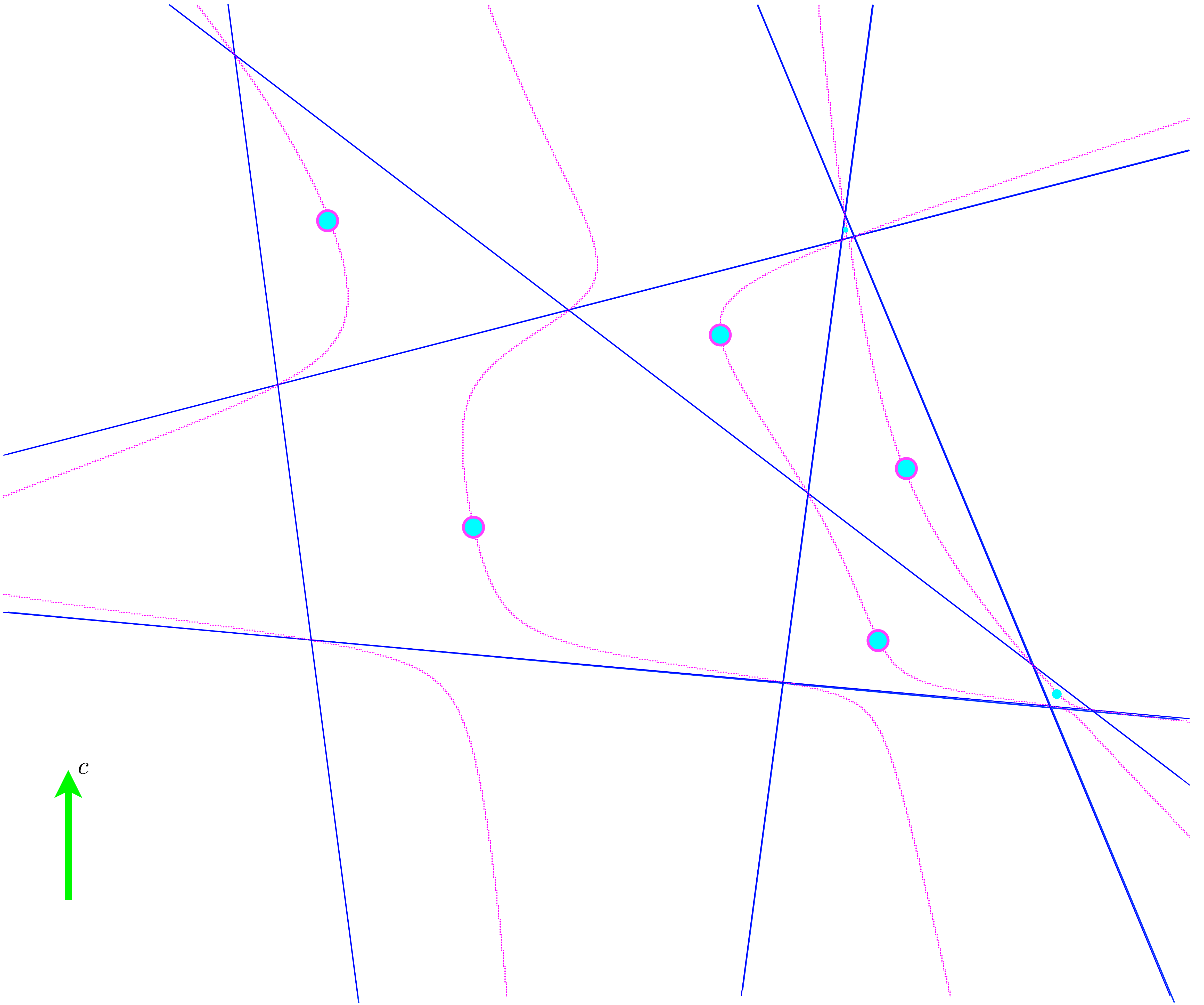
(i.e., re-center)



The straighter the central path, the better!

Mike, Jean-Pierre, Gregorio (2005):

For generic (A, b, c),
the expected Euclidean curvature of the central path
for bounded regions
does not exceed $2\pi n$.


c

Mike, Jean-Pierre, Gregorio (2005):

For generic (A, b, c),
the expected Euclidean curvature of the central path
for bounded regions
does not exceed $2\pi n$.

De Loera, Sturmfels and Vinzant (soon to be published):

For generic (b, c),
the expected Euclidean curvature of the central path
for bounded regions
does not exceed $2\pi n$
and potentially can be much better depending on A

You're great, Mike!!!