

Fast Wind Turbine Design via Geometric Programming

Warren Hoburg* and Pieter Abbeel †

UC Berkeley Electrical Engineering and Computer Science Department, Berkeley, CA, 94720, USA

Over the past two decades, applied mathematicians in the field of convex optimization have developed powerful new tools for solving certain classes of constrained optimization problems extremely reliably and efficiently. This paper introduces the (perhaps surprising) discovery that a number of prevailing physics-based models for wind turbine aerodynamics have an underlying convex mathematical structure that these new methods can exploit. In particular, they can be expressed via the functional forms that define Geometric Programs (GPs). Modern GP solvers are extremely fast even on large problems, require no initial guesses or tuning of solver parameters, and guarantee *globally* optimal solutions. In addition, for various models that cannot be manipulated into the forms required by GP, it is often possible to fit compact GP models which accurately approximate the original models. The combination of 1) fast solution methods, 2) GP-compatible aerodynamic models (presented herein), and 3) GP-compatibility of structural models (well established) makes the application of GP to large wind turbine design problems a promising approach.

Nomenclature

$(\cdot)_a, (\cdot)_t$	axial, tangential velocity components	c_l	section lift coefficient
$\Delta(\cdot)$	discretized quantity	C_P	power coefficient $\equiv \Omega Q / (\frac{1}{2} \rho V^3 \pi R^2)$
ϵ	drag to lift ratio $\equiv c_d / c_l$	F	Prandtl tip loss factor
Γ	circulation, m^2/s	f	Prandtl coordinate $\equiv \frac{B}{2} \left(\frac{R}{r} - 1 \right) \frac{W_t}{W_a}$
λ	tip speed ratio $\equiv \Omega R / V$	G	near-axis swirl correction factor
\mathbf{x}	vector of optimization decision variables	N	number of spanwise stations
Ω	rotation rate, rad/s	Q	total rotor torque, Nm
ξ	$\equiv \sqrt{\lambda^2 y^2 + 4ab} = W_t / V + s$	R	disk radius, m
ζ	non-dimensional chord $\equiv c / R$	r	spanwise coordinate, m
a	local axial induction factor $\equiv v_a / V$	s	local swirl induction factor $\equiv v_t / V$
B	number of blades	V	wind velocity, m/s
b	nondimensional axial velocity $\equiv W_a / V$	v	rotor-induced velocity at disk, m/s
c	blade chord, m	W	total velocity relative to blade
c_d	section drag coefficient	y	nondimensional spanwise coordinate $\equiv r / R$

I. Introduction

According to the World Wind Energy Association, wind turbine installations supply approximately 3% of the world's electricity consumption, with a growth rate of approximately 20% per year as of 2011. Given the wind sector's annual turnover of \$65 Billion, optimal design of wind turbines is a problem of great interest to many stakeholders. Moreover, because wind turbines must operate across a distribution of wind speeds, their design must consider performance criteria not at a single point but across a range of distinct operating conditions. This increases the number of unknown flow-field variables to be optimized over, expands problem size, and means that maintaining tractable solution times is a particularly salient consideration. Luckily, recent developments in applied mathematics can help.

*Ph.D. Student, Electrical Engineering and Computer Science, whoburg@berkeley.edu, AIAA Student Member

†Assistant Professor, Electrical Engineering and Computer Science, pabbeel@cs.berkeley.edu

Over the past two decades, *convex optimization* has grown from a specialized research discipline into the realm of a technology.¹ In particular, new computational methods and tools are capable of solving standard classes of convex optimization problems extremely reliably and efficiently. Like solving least-squares problems or linear programs^a, solving certain convex optimization problems exactly is now a straightforward task for modern solvers.

In this paper, we propose geometric programming (GP), which is one particular subclass of convex optimization problems, as a potential unifying framework for formulating and solving wind turbine design problems. We have discovered that a number of physical models common in wind turbine aerodynamic analysis can be formulated using GP-compatible objective and constraint functions. The overarching benefit that this GP formulation affords is extremely efficient and robust exact solutions via modern solvers. In addition, GP provides a number of useful mathematical guarantees, such as convergence to a *globally* optimal solution.

It should be mentioned at the outset that the GP approach is not universal: there are a number of physical phenomena and mathematical relations that cannot be expressed, even approximately, in a GP-compatible form. That said, GP has recently undergone a resurgence as researchers have discovered promising applications in a wide array of fields, including circuit design,² communication systems,³ structures,⁴ and statistics.¹

II. Background: Geometric Programming

In this section, we give an overview of the approach, define geometric programs mathematically, and describe the methods and benefits of how GPs are typically solved.

A. Conceptual Overview: The GP Design Paradigm

In MDO parlance, GP-based design is an all-at-once (AAO) approach.⁵ Every variable whose value is unknown becomes a decision variable to be determined by the optimizer. These variables include *design parameters* such as turbine disk radius, tower diameter, or airfoil thickness, along with intermediate unknown quantities, such as root bending moment. Performance criteria such as power or material cost will also be represented as unknown decision variables.

Of course, the decision variables cannot take on arbitrary values; they are constrained by physics. These physical relationships will be quantified by *constraints* on the feasible set of the GP. For practitioners using GP for design, most of the effort lies in *modeling* these physical relationships in a GP-compatible way. The details of *solving* the resulting GP are typically handled by commercial-off-the-shelf (COTS) solvers. Several open source solvers also exist.

Because every unknown is a decision variable, multi-criterion optimization becomes a simple matter of using a weighted combination of performance criteria as the objective function. The objective weightings are knobs for the designer to fine tune, or they can be varied automatically to sweep out a Pareto frontier.

B. Geometric Program: Definition

Geometric programs are constrained optimization problems where the objective and constraints are made up of *monomial* and *posynomial* functions. We will start by defining these two special function classes.

1. Monomial Functions

In geometric programming^b, a *monomial* is a function $g(\mathbf{x}) : \mathbf{x} \in \mathbb{R}_{++}^n \rightarrow \mathbb{R}_{++}$ that has the form

$$g(\mathbf{x}) = c \prod_{i=1}^n x_i^{a_i}, \quad (1)$$

where $a_i \in \mathbb{R}$, $i = 1 \dots n$, and $c \in \mathbb{R}_{++}$. For instance, the familiar expression for lift, $\frac{1}{2}\rho V^2 C_L S$, is a monomial in $\mathbf{x} = (V, C_L, S)$, with $c = \frac{1}{2}\rho$. Since the powers a_i in (1) may be negative and non-integer, expressions like

^aBoth least-squares problems and linear programs are, in fact, special cases of convex optimization problems.

^bAs noted by Boyd,⁴ the term *monomial* used in algebra is slightly different.

$\frac{x_1 x_2^{0.7} \sqrt{x_3}}{x_4}$ are also monomial functions. Since the leading coefficient c and the decision variables x_i must all be positive, the property $g(\mathbf{x}) > 0 \forall \mathbf{x}$ holds for any monomial function.

2. Posynomial Functions

A *posynomial* is a function $f(\mathbf{x}) : \mathbf{x} \in \mathbb{R}_{++}^n \rightarrow \mathbb{R}_{++}$ that has the form

$$f(\mathbf{x}) = \sum_{k=1}^K c_k \prod_{i=1}^n x_i^{a_{ik}}, \quad (2)$$

where (as before) $a_i \in \mathbb{R}$, $i = 1 \dots n$, and $c_k \in \mathbb{R}_{++}$, $k = 1 \dots K$. Thus, a posynomial is simply a sum of monomial terms, and all monomials are also posynomials (with just one term). The expression $0.23 + x_1^2 + 0.1x_1x_2^{0.8}$ is an example of a posynomial in $\mathbf{x} = (x_1, x_2)$, whereas $2x_1 - x_2^{1.5}$ is not a posynomial because negative leading coefficients c_k are not allowed.

3. Definition of a GP

A geometric program is an optimization problem of the form

$$\begin{aligned} & \text{minimize} && f_0(\mathbf{x}) \\ & \text{subject to} && f_i(\mathbf{x}) \leq 1, \quad i = 1, \dots, m, \\ & && g_i(\mathbf{x}) = 1, \quad i = 1, \dots, p, \end{aligned} \quad (3)$$

where the f_i are posynomial functions, the g_i are monomial functions, and $\mathbf{x} \in \mathbb{R}^n$ are the *decision variables*, implicitly constrained to be positive ($\mathbf{x} > 0$). In plain English, a GP minimizes a posynomial objective function, subject to monomial equality constraints and posynomial inequality constraints. Note that monomials and posynomials are closed under monomial division, so constraints of the form (posynomial \leq monomial) or (monomial = monomial) can be easily converted into the form in (3).

C. Solving Geometric Programs

Over the past two decades, technology for solving GPs has become efficient and reliable enough to make truly large-scale optimization practical. Modern *interior-point* GP solvers are approaching the efficiency of linear program (LP) solvers: they can find the global optimum of a problem with thousands of decision variables and tens of thousands of constraints on a desktop computer in minutes,¹ with additional gains if the problem is sparse.

Geometric programs inherit their desirable mathematical properties and ease of solution from the broader class of convex optimization problems to which they belong. GPs are not convex in the form (3), but a simple logarithmic change of variables transforms monomials into affine functions, and posynomials into log-sum-exp functions, both of which are convex.⁴ Aside from enabling extremely rapid solutions, convexity provides important guarantees, including

- any local optimum is also a global optimum, and
- Newton-type descent algorithms will converge to a global optimum from *any* starting point.

As a result, modern solvers do not require any ‘initial guesses’ to get started, and do not require problem-specific tuning of optimizer hyperparameters. This is a stark contrast from methods for general nonlinear optimization, which typically require initial guesses, may require the tuning of problem-specific optimizer parameters, and find local, not global, optima. Because general nonlinear methods require so many problem-specific inputs and tweaks, many practitioners end up implementing their own optimization code. GP solvers, on the other hand, are robust and general enough for designers to confidently leave the optimization process to standard software packages.

III. Wind Turbine Aerodynamics

We adopt the aerodynamic models described by Drela,⁶ which are an extension of the classical blade-element/vortex formulation originally developed by Betz, Goldstein,⁷ Theodorsen,⁸ and Larrabee.⁹

A. Flow-field Velocities

This section defines the variables that parameterize the velocity field around the rotor. The turbine span-wise radial coordinate $y \equiv r/R$ ranges from 0 at the root to 1 at the tip, and the angular rotation rate Ω creates a tip speed ratio $\lambda \equiv \frac{\Omega R}{V}$. The flow field model assumes that the far-field velocity V superimposes with 1) the blade rotational speed Ωr , and 2) the turbine induced velocity $v(r)$, creating a radially-varying relative velocity W at the turbine blade.

Each velocity can be decomposed into axial $(\cdot)_a$ and tangential $(\cdot)_t$ components. We also define (radially-varying) nondimensional factors for axial induction ($a \equiv v_a/V$), axial velocity at the disk ($b \equiv W_a/V$), and tangential (swirl) induction ($s \equiv v_t/V$). This results in the following relations among the flow-field velocity components, expressed in both dimensional and non-dimensional forms:

$$W_a = V - v_a \qquad b = 1 - a \qquad (4)$$

$$W_t = \Omega r + v_t \qquad \frac{W_t}{V} = \lambda y + s \qquad (5)$$

B. Induced Velocity Relations

Following Drela,⁶ we assume that at any radial coordinate, the induced velocity v is perpendicular to the local flow velocity W :

$$v_a W_a = v_t W_t \qquad ab = s(\lambda y + s) \qquad (6)$$

Note that (6) is a quadratic equation, which can be solved for v_t and W_t (or s and W_t/V):

$$v_t = \frac{-\Omega r}{2} + \frac{\sqrt{\Omega^2 r^2 + 4v_a W_a}}{2} \qquad s = \frac{-\lambda y}{2} + \frac{\sqrt{\lambda^2 y^2 + 4ab}}{2} \qquad (7)$$

$$W_t = \frac{\Omega r}{2} + \frac{\sqrt{\Omega^2 r^2 + 4v_a W_a}}{2} \qquad \frac{W_t}{V} = \frac{\lambda y}{2} + \frac{\sqrt{\lambda^2 y^2 + 4ab}}{2} \qquad (8)$$

The induced swirl velocity v_t is assumed to be related to the local circulation Γ by

$$B\Gamma = 4\pi r v_t F G \qquad (9)$$

$$G \equiv \sqrt{1 + \left(\frac{4W_a}{\pi B W_t}\right)^2}, \qquad (10)$$

where B is the number of blades, F is a Prandtl tip loss correction factor (discussed later), and G is a correction factor that becomes significant close to the axis.

Finally, the circulation is related locally to the blade section chord and lift coefficient:

$$\Gamma = \frac{1}{2} W c c_l \qquad (11)$$

C. Torque and Power

At each span-wise coordinate r , the blade cross section airfoil exerts a lift force perpendicular to the local flow velocity W , and a drag force parallel to W . The resulting differential power created by each annular portion of the radius is

$$\Omega dQ = \rho B \Gamma (W_a - \epsilon W_t) \Omega r dr, \qquad (12)$$

where $\epsilon = c_d/c_l$ is the local drag-to-lift ratio. Substituting the Helmholtz relation (9), we can express the differential power in terms of the coefficient $C_P \equiv \Omega Q / \frac{1}{2} \rho V^3 \pi R^2$,

$$\frac{dC_P}{dy} = 8s\lambda y^2 F G (b - \epsilon(\lambda y + s)) \qquad (13)$$

IV. GP-compatible Formulation

We now reformulate the preceding aerodynamic models into a GP-compatible posynomial constraint set.

A. GP-compatible Aerodynamic Relations

Substituting (6) into (13), the differential power is bounded from above by the posynomial constraint

$$sb \geq \frac{dC_P}{dy} \frac{1}{8\lambda y^2 FG} + \epsilon ab \quad (14)$$

The swirl velocity s is constrained by (6). Substituting (5) and (8), this can be manipulated into a set of posynomial constraints:

$$2ab \geq s\lambda y + s\xi \quad (15)$$

$$\xi^2 \geq \lambda^2 y^2 + 4ab \quad (16)$$

It is straightforward to show that $V\xi = V\sqrt{\lambda^2 y^2 + 4ab} = W_t + v_t$. Note that we could have alternatively formulated a simpler posynomial constraint based on (6) alone, but the formulation above is chosen since ξ will be useful later.

Finally, (4) is equivalent to the posynomial constraint

$$1 \geq a + b \quad (17)$$

B. Aside: Relation to Actuator Disk Theory and the Betz Limit

Before we consider additional models, it is worth studying the limiting behavior of the constraints defined thus far. If we ignore the correction factors ($F = G = 1$) and let the viscous drag vanish ($\epsilon \rightarrow 0$), then (14), (15), and (16) reduce to

$$dC_P \leq \frac{16ab^2}{1 + \sqrt{1 + \frac{4ab}{\lambda^2 y^2}}} y dy \quad (18)$$

It is thus clear that to maximize dC_P in the absence of viscous drag, we should allow the tip speed ratio to become large ($\lambda \rightarrow \infty$), which causes the denominator to approach 2, independent of y . This corresponds to the light loading limit of actuator disk theory. We are then left with the problem

$$\begin{aligned} &\text{maximize} && C_P = \int_0^1 8ab^2 y dy \\ &\text{subject to} && 1 \geq a + b \end{aligned}$$

The solution is $a = 1/3$, $b = 2/3$, $C_P = 16/27$, which is the famous Betz limit.

C. Spanwise Discretization

In the general case, viscous drag makes large tip speed ratios impractical, and all the flowfield quantities vary with the spanwise coordinate y . To determine the total rotor power, we break the span into N discrete segments, replacing dC_P/dy with $\Delta C_P/\Delta y$. Each of the flowfield quantities (a , b , s , ξ , F) becomes a vector variable with one entry per spanwise station, and (14), (15), (16), and (17) are imposed at each station.

One natural (and popular) choice for the discretization is bins of equal width $\Delta y = 1/N$, with the stations y_i prescribed to lie (for example) at the midpoint of each bin. Under this scheme, each spanwise element produces a different discrete power $(\Delta C_P)_i$, with the total power given by

$$C_P = \sum_{i=1}^N (\Delta C_P)_i \quad (19)$$

The difficulty with this formulation is that according to (19), C_P is bounded from *below* by a posynomial, leaving C_P unbounded above if we seek to maximize it. Indeed, *maximizing* a sum of convex functions is known to be a difficult, non-convex optimization problem in general.

We propose an alternative formulation, which circumvents the sum maximization issue. Instead of defining each spanwise bin to have the same Δy , we allow y and Δy to be (vector) variables, but require that each bin produce the same differential power $\Delta C_P = C_P/N$. Maximizing C_P then corresponds to maximizing the scalar ΔC_P (which in turn corresponds to minimizing its inverse).

Because the bin positions and sizes are now variables, we must constrain them to fit inside the rotor radius without overlap. We also constrain the root chord so as to avoid overlap of the blades. Assuming that each station lies at the midpoint of its bin, we have the posynomial constraint set:

$$1 \geq y_N + \frac{(\Delta y)_N}{2} \quad (20)$$

$$y_{i+1} \geq y_i + \frac{(\Delta y)_i}{2} + \frac{(\Delta y)_{i+1}}{2}, \quad i = 1 \dots (N-1) \quad (21)$$

$$y_1 \geq \frac{B\zeta_1}{2\pi} + \frac{(\Delta y)_1}{2} \quad (22)$$

Thus, by choosing a computational mesh with uniform power output instead of uniform spacing, one obtains a GP-compatible formulation. Note that the power basis need not be uniform, so long as we define it a priori. For example, to improve the fidelity of inboard structural analysis (where moments are large), we might choose a basis with smaller power bins near the root.

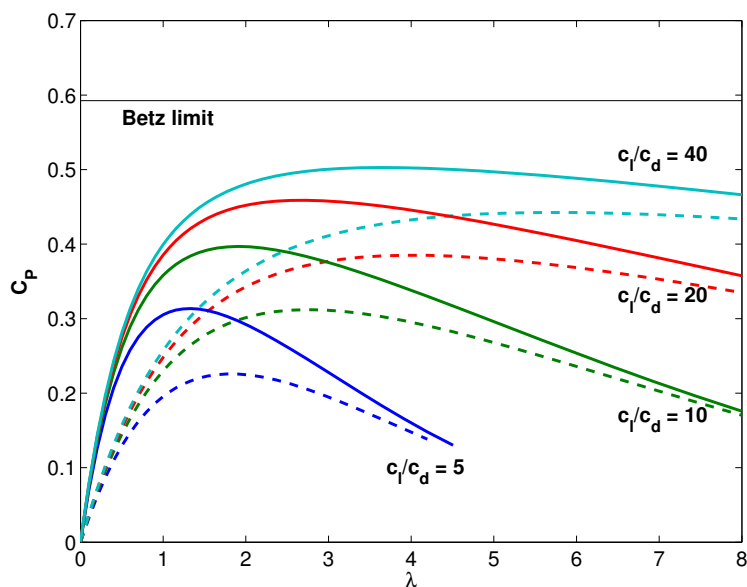


Figure 1. Maximum total power coefficient C_P as a function of tip speed ratio λ and lift to drag ratio $1/\epsilon$. Solid lines correspond to a model for induced wake rotation that does not account for tip loss effects, whereas dashed lines include a Prandtl tip loss correction. Note the flattening of the optimum and right-shift associated with tip losses. Local flow conditions were modeled at a total of 10 span-wise stations. Each point $C_P(\lambda, \epsilon)$ represents the solution to a GP with 60 variables and 50 posynomial inequality constraints, solved in less than 0.1 sec on a standard desktop computer.

D. Prandtl Tip Loss Correction

Just as a wing of finite span experiences induced drag due to tip vortices, a rotor blade of finite radius exhibits reduced lift (and therefore reduced power production) near the tip. One common model^{6,10} for this effect is the Prandtl correction factor, F :

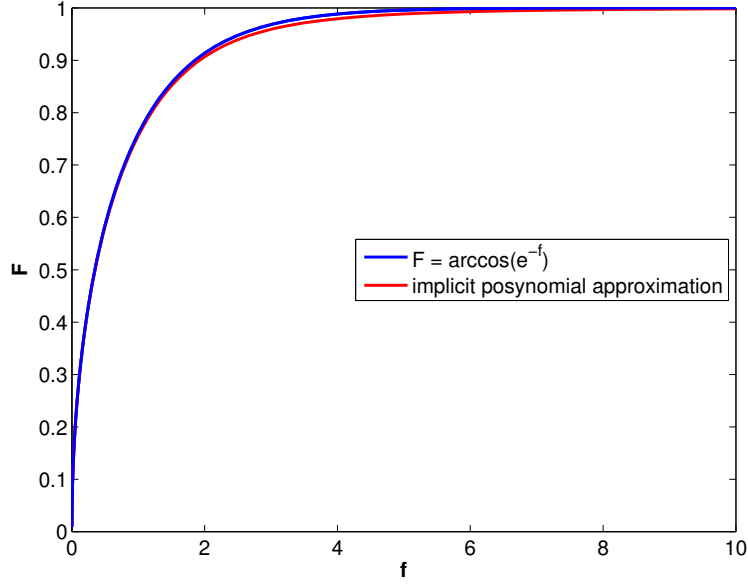


Figure 2. The standard Prandtl tip loss correction factor, $F = \cos^{-1}(e^{-f})$, is well-approximated by an implicit posynomial constraint, $1 \geq F^{1.56} + 1.655F^{5.51}/f^{2.76}$, which is compatible with geometric programming.

$$f \equiv \frac{B}{2} \left(\frac{R}{r} - 1 \right) \frac{W_t}{W_a} \quad (23)$$

$$F = \frac{2}{\pi} \arccos(e^{-f}), \quad (24)$$

This model is not directly compatible with GP due to the $\arccos(\exp(\cdot))$ term. As shown in Figure 2, however, the model is well-approximated by the following set of posynomial inequality constraints:

$$\frac{1}{y} \geq 1 + \frac{2fs}{Ba} \quad (25)$$

$$1 \geq F^{1.56} + 1.655 \frac{F^{5.51}}{f^{2.76}} \quad (26)$$

The effect of tip loss on total power production appears in Figure 1.

E. Near-axis correction factor

The near axis correction factor G is not compatible with geometric programming. Although the expression (10) can be expressed as a posynomial, imposing such a constraint would bound G from *below*. Since (14) loosens with increasing G , any model for G must bound it from *above*.

For now, we adopt the conservative approximation

$$G \leq 1, \quad (27)$$

which will cause us to underestimate power production by approximately 1% for a typical design problem.

F. Blade Profile Drag

The drag-to-lift ratios ϵ are in general complicated functions of local lift coefficient, Reynolds number, Mach number, and airfoil geometry. One common approach, which turns out to be GP-compatible, is to model ϵ for a given airfoil as the product of a quadratic function of lift coefficient with a power law scaling on

Reynolds number.⁶ There is also room for higher fidelity models that remain GP compatible. For example, it has recently been shown that for certain airfoil families, $c_d(c_l, \text{Re}, \tau)$ is well-approximated by a posynomial function over large parameter ranges of interest.¹¹

G. Tower Loads

The thrust T exerted by the rotor creates stresses in the tower, and therefore affects tower sizing and material costs. One design case of interest is the wind turbine with the maximum shaft power for a given tower load. This corresponds to the *minimum induced loss* operating condition.⁶ At a given radius, the differential thrust per unit torque is governed by

$$\eta = \frac{V}{\Omega} \frac{dT}{dQ} = \frac{V}{\Omega r} \frac{W_t + \epsilon W_a}{W_a - \epsilon W_t}. \quad (28)$$

Introducing $\lambda_r = \lambda r/R$ and a helper variable S , one can obtain the following set of equivalent GP-compatible constraints governing thrust as a function of power production:

$$SV \frac{dT}{dP} \geq 1 + a' + \epsilon \lambda_r \frac{a'}{a} + \epsilon \lambda_r \frac{a'^2}{a} \quad (29)$$

$$1 \geq S + a + \epsilon \lambda_r + \epsilon \lambda_r a' \quad (30)$$

Tower loads are therefore readily considered within the GP design framework. For example, one can formulate posynomial models for tower structural sizing and material costs as a function of loads, thus enabling simultaneous design optimization of rotor and tower and fast exploration of the power - material cost Pareto frontier.

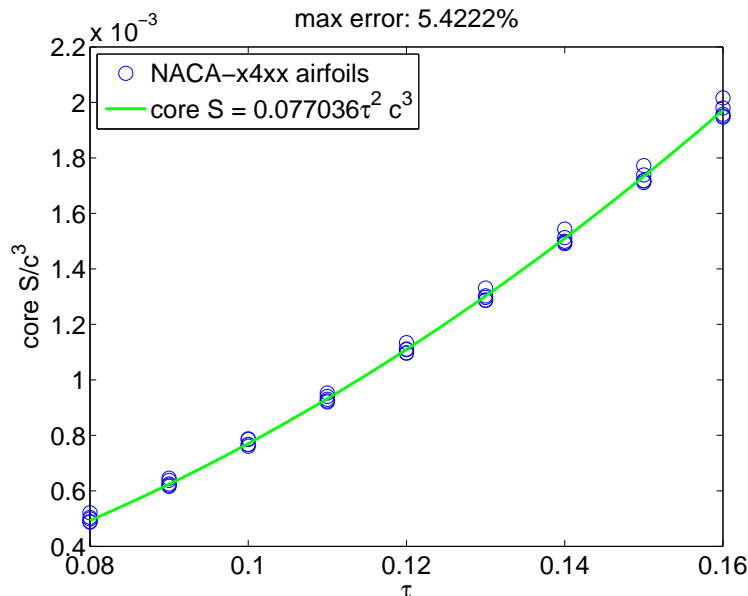


Figure 3. A simple GP-compatible model for root section modulus S_{xx} of NACA-x4xx series airfoils, as a function of the thickness ratio τ . The blue data points were calculated using Green's theorem¹² for five different values of camber ranging from 0–4%. The green line represents a GP-compatible monomial fit to the data. This model enables structural sizing of the blade cross section with respect to stress limits. A similar model for I_{xx} enables consideration of deflection constraints (e.g. to avoid tower strike).

V. Concluding Remarks

We have described the discovery that seemingly complicated relationships which govern wind turbine aerodynamics have an underlying convex structure that can be expressed in terms of a geometric program. This observation has several important consequences. Chiefly among them, it:

- improves understanding of problem structure. For example, this discovery explains why typical Newton-step based rotor analysis codes are observed to converge in very few iterations,⁶ and gives one confidence that convergence will be robust to choice of initialization.
- enables extremely large problem instances to be solved reliably and quickly. For example, multidisciplinary aero-structural design with respect to a probability distribution of operating conditions tends to create extremely large problem instances, but the speed of modern GP solvers puts reasonable solution times within reach.

Acknowledgments

This work is supported by an NSF Graduate Research Fellowship, an Alfred P. Sloan Research Fellowship, a gift from Intel and a gift from Toyota.

References

- ¹Boyd, S. and Vandenberghe, L., *Convex Optimization*, Cambridge University Press, New York, NY, USA, 2004.
- ²Boyd, S. P., Jean Kim, S., Patil, D. D., and Horowitz, M. A., “Digital Circuit Optimization via Geometric Programming,” *Operations Research*, Vol. 53, 2005, pp. 899–932.
- ³Chiang, M., “Geometric programming for communication systems,” *Commun. Inf. Theory*, Vol. 2, July 2005, pp. 1–154.
- ⁴Boyd, S., Kim, S.-J., Vandenberghe, L., and Hassibi, A., “A tutorial on geometric programming,” *Optimization and Engineering*, 2007.
- ⁵Cramer, E. J., J.E. Dennis, J., Frank, P. D., Lewis, R. M., and Shubin, G. R., “Problem Formulation for Multidisciplinary Optimization,” *Presented at the AIAA Symposium on Multidisciplinary Design Optimization*, August 1993.
- ⁶Drela, M., “QPROP Formulation - Theory Document,” 2006.
- ⁷Goldstein, S., “On the vortex theory of screw propellers,” *Proceedings of the Royal Society of London. Series A*, Vol. 123, No. 792, 1929, pp. 440–465.
- ⁸Theodorsen, T., *Theory of propellers*, Vol. 9, McGraw-Hill Book Company, 1948.
- ⁹Eugene Larrabee, E. and French, S. E., “Minimum induced loss windmills and propellers,” *Journal of Wind Engineering and Industrial Aerodynamics*, Vol. 15, No. 1, 1983, pp. 317–327.
- ¹⁰Manwell, J., McGowan, J., and Rogers, A., *Wind Energy Explained: Theory, Design and Application*, John Wiley & Sons, 2010.
- ¹¹Hoburg, W. and Abbeel, P., “Geometric Programming for Aircraft Design Optimization,” *53rd AIAA Structures, Structural Dynamics, and Materials Conference*, 2012.
- ¹²Hoburg, W., *Aircraft Design Optimization as a Geometric Program*, Master’s thesis, UC Berkeley, 2011.