Differential Geometric Foundations for Power Flow Computations

Franz-Erich Wolter, Benjamin Berger, Alexander Vais
Leibniz Universität Hannover, Germany

Abstract—This paper aims to systematically and comprehensively initiate a foundation for using concepts from computational differential geometry as instruments for power flow computing and research. At this point we focus our discussion on the static case, with power flow equations given by quadratic functions defined on voltage space with values in power space; both spaces have real Euclidean coordinates. The central issue is a differential geometric analysis of the power flow solution space boundary (SSB, also in a simplifying way, called saddle node bifurcation set, SNB) both in voltage and in power space. We present different methods for computing tangent vectors, tangent planes and normals of the SSB and the normals’ derivatives. Using the latter we compute normal and principal curvatures. All this is needed for tracing the orthogonal projection of points on curves in voltage or power space onto the SSB on points closest to the given points on the curve, thus obtaining estimates for their distance to the SSB. As another example how these concepts can be useful, we present a new high precision continuation method for power flow solutions close to and on the SSB called local inversion of the power flow map from voltage to power space, assuming the dimension of power flow’s Jacobean zero space, called KERNEL, is one. For inversion, we present two different geometry-based splitting techniques with one of them using the aforementioned orthogonal tracing method. The other considers the power flow map close to the SSB as a perfect quadratic folding construction. Here the singular quadratic folding part is merely restricted to one-dimensional kernel spaces mapping the latter to euclidean rays. The inversion is then achieved by an unfolding construction geometrically inverting the prior folding. Here accuracy of the unfolding is benefiting from our splitting construction in a meta sense restricting the singular part of the inversion essentially to a one-dimensional real square root operation. We sketch basic results on the local topology of the SSB and via topological analysis disprove the existence of fork type branching in planar sections of the SSB, that were numerically observed in a major report. Finally we indicate the relevance of geometric analysis of the power flow solution space boundary has developed tools for subtle and precise computations on those submanifolds [1]; those results also have engineering implications for the problem at hand, which we intend to elaborate [2]–[6]. In power flow computing there seem to be no systematic earlier attempts to do so.

An AC power transmission system is usually modeled as a graph. The vertices of the graph are called buses and represent generators or consumers. The edges are weighted with complex numbers and represent transmission lines with associated admittances. The state of an alternating current power transmission system can approximately be captured by two sets of variables indexed by the set of buses. Complex variables are usually split in their real and their imaginary part in order to obtain pure real formulations presented in Euclidean coordinates resulting in quadratic equations giving several special properties which can be exploited [7]. This presentation being extensively used in research in power flow computing is very convenient for presenting our research and methods. Therefore we use it here. It is possible to abstract from the dynamical behavior of the system [8].

This problem is presented as an analysis of the so-called static power flow equation, as described in a seminal paper of Hiskens [9]. Even the latter seemingly special problem is extremely complex and topic of an awesome amount of ongoing research.

Following a presentation in [9] we look at this problem in a notationally generalized setting as we want to keep the notations and descriptions simple and short. Thus we have only the voltage space $V = \mathbb{R}^n$, with vectors $v \in V$ and the power space $P = \mathbb{R}^n$, with vectors $p \in P$ and the so called power flow map $F : V \to P$ given by quadratic forms. All the engineering relevant requirements may then be obtained
by considering particular coordinates for vectors in $V$ and $P$-space, say referring to features such as active load and power injection, active and reactive coordinates in power space. Of course there are also additional nonlinear constraints such as those for voltage magnitudes and additional ones that may e.g. result from investigating special situations. Using the typically employed complex notation or employing various different types of variables would confuse the explanation of our methods as those are independent of the special interpretation of the coordinates involved.

In power flow applications, the function $F$ is not just any quadratic function, but has a symmetry that allows to declare one bus as the “slack bus”, that is to set one voltage phasor to any nonzero value, typically 1, without loss of generality. This eliminates two variables, but we lose the property that the functions under consideration are homogeneous. We chose to not make use of this symmetry and to formulate our equations with respect to arbitrary homogeneous polynomials.

For explaining our concepts we focus on some major fundamental aspects of problems causing basic difficulties in many sample problems in power flow computing. Once the formulas are available they may be applied to particular equations stemming from power flow computing. We did this for sample problems involving concrete sample networks with up to 220 dimensions, taken from [10].

In some technical and partly differential geometric aspects works in power flow computing described in [11]–[14] are close to ours. Albeit restricted to the computationally more accessible power space, these works use and benefit from differential geometric concepts for computing nearest points on and distance estimations to saddle point bifurcations. There are major differences in our computations: We present distance and nearest point computations including the harder case are major differences in our computations: We present distance accessible power space, these works use and benefit from the formulas are available they may be applied to particular fundamental aspects of problems causing basic difficulties with respect to arbitrary homogeneous polynomials.

This eliminates two variables, but we lose the property that the functions under consideration are homogeneous. We chose to not make use of this symmetry and to formulate our equations with respect to arbitrary homogeneous polynomials.

For explaining our concepts we focus on some major fundamental aspects of problems causing basic difficulties in many sample problems in power flow computing. Once the formulas are available they may be applied to particular equations stemming from power flow computing. We did this for sample problems involving concrete sample networks with up to 220 dimensions, taken from [10].

In some technical and partly differential geometric aspects works in power flow computing described in [11]–[14] are close to ours. Albeit restricted to the computationally more accessible power space, these works use and benefit from differential geometric concepts for computing nearest points on and distance estimations to saddle point bifurcations. There are major differences in our computations: We present distance and nearest point computations including the harder case on saddle point bifurcations. Recent works such as [15] on finding feasible solution sets points of maximal and minimal distance to SSB indicates current interest in distance computations of operation points to the SSB. [15] searching in feasible solution sets points of maximal or minimal distance to SNB of SSB proves the need for distance computations of operation points to the SSB. Oppose to us, [15] seems not to use of maximal principal curvature—or equivalent second-order surface properties of the SSB, needed for local distance minimality (cf. fig. 4). This would require observing geometric second-order input from both SNB and the feasible manifold.

Although the question of systematically solving the power flow equation close to the singular boundary is considered important, we did not see works employing an approach similar to our local inversion technique, which separates the difficult-to-invert part into a one-dimensional subspace.

Discussing applications of our new methods on special EE sample problems is so far still in the beginning and have not been our focus, as we are still developing and improving basic tools. We also feel that the more global vision of our concepts outlined with some details and many figures in our work [16] is more important and with details and proofs should have a broader long time impact. In [16] we prove, discuss and illustrate basic results of the local topological SSB structure, being here only briefly sketched. Likewise here we only descriptively indicate the relevance of geodesic coordinates for constraint solution sets and refer for details to [16].

II. GEOMETRIC ENTITIES ON THE SSB

The set of points where the Jacobian of the power flow map is singular is called the solution space boundary (SSB) or bifurcation set [9]. Its physical relevance is that the solution of the power flow equation $F(v) = p$ cannot be continued continuously beyond the SSB, leading to instability of the system. Understanding the SSB is important in order to keep a safe distance from it.

The SSB normal in power space, $N_P$, at a point is simply the left eigenvector of the differential for the eigenvalue 0:

$$ (D F) \approx 0 $$(1)

$$ N_P \approx 1 $$ (2)

The columns of (D F) are all orthogonal to the normal, so they span the tangent space.

The vector $N_P$ is in the zero space of (D F) $\approx$, which we call kernel. We mostly have the generic case where the dimension of this zero space is 1, allowing us to identify it with a non-oriented unit vector, called kernel vector. Because $N_P$ is in the kernel of (D F) $\approx$, we will also call it $\hat{k}$, whereas $k$ will be the kernel vector of D F, thus being much harder to compute than $N_P$.

In voltage space, things are more complicated. Because the SSB is an iso-surface where $\lambda_0$, the smallest eigenvalue of (D F), is zero, the unit normal in voltage space $N_V$ is collinear with the gradient of $\lambda_0$.

Differentiating the equations expressing that $(\lambda_0, \hat{k})$ is an eigenpair with respect to a voltage variable $v_i$ at $\lambda = 0$, we obtain a linear equation system with unknowns $d\lambda_0$ and $d\hat{k}$:

$$ (D F) \approx d \frac{\hat{k}}{d v_i} = \frac{d \lambda_0}{d v_i} - d \frac{(D F) \approx}{d v_i} \frac{\hat{k}}{k} $$

Thus we obtain both the (unnormalized) normal vector in voltage space and the derivatives of the normal vector in power space (provided the rank of D F is $n-1$ and hence the zero eigenvalue is simple). We are also interested in curvatures of the SSB in voltage space. So in order to get the derivatives of the unnormalized normal in voltage space, we need to differentiate again with respect to a voltage direction $v_i$:

$$ (D F) \approx \frac{d^2 \hat{k}}{d v_i d v_j} = \frac{d^2 \lambda_0}{d v_i d v_j} - \frac{d^2 (D F) \approx}{d v_i d v_j} \frac{\hat{k}}{k} - \frac{d (D F) \approx}{d v_i} \frac{d \hat{k}}{d v_i} \frac{d \lambda_0}{d v_i} \frac{\hat{k}}{k} $$

Within this context the following equation is derived:

$$ (D F) \approx \frac{d^2 \hat{k}}{d v_i d v_j} = \left( \frac{d (D F) \approx}{d v_i} \frac{d \hat{k}}{d v_i} \frac{d \lambda_0}{d v_i} \frac{\hat{k}}{k} \right) + \frac{d \hat{k}}{d v_i} \frac{d \lambda_0}{d v_i} + \frac{d \hat{k}}{d v_i} \frac{d \lambda_0}{d v_i} $$

Thus we obtain both the (unnormalized) normal vector in voltage space and the derivatives of the normal vector in power space (provided the rank of D F is $n-1$ and hence the zero eigenvalue is simple). We are also interested in curvatures of the SSB in voltage space. So in order to get the derivatives of the unnormalized normal in voltage space, we need to differentiate again with respect to a voltage direction $v_i$:
By inserting the previously calculated values of \( \frac{d^2\lambda_0}{dv_i} \) and \( \frac{d^{\lambda_0}}{dv_i dv_j} \), we get a linear equation system for the second derivatives \( \frac{d^2\lambda_0}{dv_i dv_j} \) and \( \frac{d^2\lambda_0}{dv_i dv_j} \). Since this equation system has the same matrix as (3) (and the same as the matrices needed for higher derivatives), regardless of \( i \) and \( j \), it pays off to invert the matrix once. Once the matrix has been inverted, derivatives of the same order can be computed in parallel.

This is a systematic approach to compute arbitrary derivatives of the normal vectors. However, in our actual numerical experiments where we only needed the normals and their first derivatives, we used ideas presented in the following which allowed us to obtain the first derivatives of the normals without using derivatives of eigenvalues and eigenvectors.

In our implementation, we made no use of derivatives of eigenvalues and eigenvectors. In experiments where we only needed the normals and their derivatives, regardless of normal, see [17].

First derivatives of the normal are needed to define normal curvatures in various tangential directions \( \dot{\epsilon} \). Let \( N \) be the unit normal vector in either power or voltage space. The normal curvature \( \kappa_N(\dot{\epsilon}) \) in the unit direction \( \dot{\epsilon} \) is defined as

\[
\kappa_N(\dot{\epsilon}) = W(\dot{\epsilon}) \cdot \dot{\epsilon},
\]

where \( W \) is the negative differential of the unit normal, \( W = -D\,N \), called Weingarten map or shape operator.

If \( \dot{\epsilon} \) is actually the tangent vector of an arc-length parametrized surface curve \( c \), then using Meunier’s theorem there is a simpler way to compute the normal curvature:

\[
\kappa_N(\dot{\epsilon}) = N \cdot \ddot{\epsilon}.
\]

Only the component of \( \dddot{\epsilon} \) perpendicular to the surface is relevant in (6), and this normal component only depends on \( \dot{\epsilon} \) and no other details of the curve.

We can use this for computing the shape operator without the need to differentiate the normal vector via the subsequent way, inspired by [18]: Choose \( \frac{n^2-n}{2} \) suitable unit tangent directions \( \dot{\epsilon}_i \). Since the curves \( \dot{\epsilon}_i \), assumed to be parametrized proportional to arc length, run inside the surface, the normal component of each \( \dddot{\epsilon}_i \) is uniquely determined by \( \dot{\epsilon} \) (see (13)) and can be used in (6) to compute the normal curvatures in the directions \( \dot{\epsilon}_i \). By inserting all the known normal curvatures into \( \frac{n^2-n}{2} \) instances of (5), one instance for each \( i \), we get a linear equation system for the components of the matrix representation of \( W \):

\[
W(\dot{\epsilon}_i) \cdot \dot{\epsilon}_i = \kappa_N(\dot{\epsilon}_i).
\]

This is an equation system with \( \frac{n^2-n}{2} \) equations and the same number of unknowns; \( W \) is self-adjoint with respect to the Riemannian metric tensor or first fundamental form \( g \), so only \( \frac{n^2-n}{2} \) entries are needed to determine it. The metric tensor is used to define the inner products and thereby lengths and angles of tangential vectors. In our case, the manifold is embedded in a Euclidean space and so \( g \) is simply the restriction of the ambient Euclidean metric to the tangent space: \( g(\dot{\epsilon}_i, \dot{\epsilon}_j) \) is simply \( g_{ij} := \dot{\epsilon}_i \cdot \dot{\epsilon}_j \). In particular \( W_{ik} = -\sum_j g^{ij} L_{jk} \) where the \( g^{ij} \) are the components of the inverse of the matrix \( (g_{ij}) \). The \( L_{jk} \) are the components of the second fundamental form, which form a symmetric matrix.

By cleverly choosing the directions \( \dot{\epsilon}_i \), we can ensure that the equation system is sparse and efficiently solvable: Choose the first \( n-1 \) directions as the standard basis of the tangent space, and the remaining directions as the sums of each pair of distinct basis vectors. The equation system for the components of \( L \) then consists of the equations:

\[
L_{ii} = \kappa_N(\dot{\epsilon}_i) \quad \text{for } 1 \leq i < n
\]

\[
L_{ii} + 2L_{ij} + L_{jj} = \kappa_N(\dot{\epsilon}_i + \dot{\epsilon}_j) \quad \text{for } 1 \leq i < j < n.
\]

Since \( \kappa_N \) is applied to non-unit vectors here, we must define it to behave as a quadratic form: \( \kappa_N(sv) = s^2 \kappa_N(v) \).

It remains to find a possible second derivative \( \dddot{\epsilon} \) (we’re only interested in the normal part) of an arc-length parameterized curve \( c \) going in the direction \( \dot{\epsilon} \), let us first consider some algebraic properties of the Jacobian of a quadratic function \( F: \mathbb{R}^n \rightarrow \mathbb{R}^n \). Such a function can always be written using symmetric matrices \( A_i \), \( 1 \leq i \leq n \):

\[
F(v) = \left( \begin{array}{c} F_1(v) \\ \vdots \\ F_n(v) \end{array} \right) = \left( \begin{array}{c} v^\top A_1 v \\ \vdots \\ v^\top A_n v \end{array} \right),
\]

Because these matrices are symmetric, the Jacobian is simply:

\[
D \, F(v) = 2 \cdot \left( \begin{array}{c} v^\top A_1 \\ \vdots \\ v^\top A_n \end{array} \right)
\]

and one easily checks that

\[
(D \, F(x))v = (D \, F(v))x.
\]

The Hessian of a quadratic form is constant:

\[
H \, F(x) = \frac{dD \, F(x)}{dx} = 2 \cdot \left( \begin{array}{c} A_1 \\ \vdots \\ A_n \end{array} \right).
\]

This implies the following:

\[
v^\top (H \, F(x)) = (D \, F)(v).
\]

With this, we can obtain the second derivatives of the curves \( c \) in voltage space by solving the subsequently derived linear systems (11), (12). For this with \( \tilde{k} \) being the kernel vector of \( D \, F \), we differentiate \( (D \, F(c)) \tilde{k} = 0 \) twice, applying (10) and (9):

\[
(D \, F(k))\ddot{c} + (D \, F(c))\dddot{k} = 0. \tag{11}
\]

\[
(D \, F(k))\ddot{c} + (D \, F(c))\dddot{k} = -2(D \, F(c)) \dddot{k}. \tag{12}
\]

This is an under-determined equation system for the unknowns \( \dddot{c} \) and \( \dddot{k} \). Hence we make it uniquely solvable by requiring the
kernel to be normalized (differentiating \( k \cdot k = 0 \) twice gives \( \dot{k} \cdot k = -\hat{k} \cdot \hat{k} \)) and assuming, since we’re only interested in the normal part of \( \dot{c} \), that \( \dot{c} = \kappa N_v \) for some \( \kappa \in \mathbb{R} \). Finally, we arrive at an equation system with \( n + 1 \) unknowns and the same number of equations:

\[
\begin{pmatrix}
  (D F(k)) N_v & D F(c) \\
  0 & k^\top
\end{pmatrix}
\begin{pmatrix}
  \kappa \\
  \dot{k}
\end{pmatrix}
= \begin{pmatrix}
  -2(D F(\dot{c})) \hat{k} \\
  -\hat{k} \cdot \hat{k}
\end{pmatrix}.
\tag{13}
\]

To determine \( \dot{k} \), we add the condition that \( k \) remain normalized, expressed as \( k \cdot \dot{k} = 0 \), to Eq. (8):

\[
\begin{pmatrix}
  D F(c) \\
  k^\top
\end{pmatrix} \dot{k} = \begin{pmatrix}
  -(D F(k)) \dot{c} \\
  0
\end{pmatrix}.
\tag{14}
\]

These are \( n + 1 \) equations for \( n \) unknowns, but they always admit a solution since the first \( n \) equations are linearly dependent.

In our specific setting, this completes the computation of normal curvatures for curves on the SSB, given their tangent vectors. In the preceding computations and considerations related to computing \( W \) or equivalently \( D N_v \) (starting with (5)), we were using the assumption that normal curvatures for tangent directions are known. Combining these results then we can now concretely compute \( D N_v \) and \( W \) for the SSB in voltage space. We used this method in our implementation for all numerical experiments involving \( D N_v \) and \( W \).

The second-derivatives of the images of curves on the image of the SSB in voltage space require slightly more effort. For this, we use the fact that the first derivative of the image \( F(c) \) with respect to the curve parameter \( t \) is \( (D F(c)) \dot{c} \). If we differentiate this again and then apply (10), we have

\[
\frac{d^2 F(c)}{dt^2} = \frac{d(D F(c)) \dot{c}}{dt} = (D F(\dot{c})) \ddot{c} + (D F(c)) \ddot{c}.
\]

When \( n \) is large and we are interested in only a few normal curvatures in power space, say one in direction \( (D F(c)) \dot{c} \), it is more appropriate to not compute the complete shape operator using (8), but rather use the following basic classical formula for applying the shape operator, given in the local coordinates provided by interpreting \( F \) as parameterization of power space by voltage space:

\[
W(\dot{c})_i = g^{-1} \tilde{L} \dot{c},
\tag{15}
\]

restricted to the tangent space of the SSB image in power space. Here \( g = (D F)^\top (D F) \) is the first fundamental form related to \( F \), which is given by a sparse symmetric matrix. The matrix \( \tilde{L} \) yields the second fundamental form of the SSB-image when restricted to tangential space. Its components are \( \tilde{L}_{ij} = \frac{\partial^2 p}{\partial x_i \partial x_j} \cdot N_p \), which is also a sparse symmetric matrix. The matrix \( g \) is singular on the SSB. But we can avoid having to invert the singular \( g \) by multiplying it to the left side:

\[
g(W(\dot{c})) = \tilde{L} \dot{c}.
\tag{16}
\]

To make this linear equation system for \( W(\dot{c}) \) more palatable to numerical solvers, we may want to ensure the right hand side does not have a component in the unit direction \( k \) of the kernel of \( g \) by writing

\[
(g + \varepsilon k k^\top)(W(\dot{c})) = (\varepsilon - k k^\top) \tilde{L} \dot{c}.
\tag{17}
\]

for some \( \varepsilon > 0 \). The solution \( W(\dot{c}) \) may not be tangential in voltage space, but since we are only interested in its image \( (D F)(W(\dot{c})) \) in power space, this does not matter.

Since in our context \( g \) originates from the adjacency matrix \( D F \) of a mostly planar graph with node degree in practice bounded by some small \( k \in \mathbb{N} \), we suggest solving this equation system using an algebraic multigrid technique. This should take \( O(nk^2) \) time.

We will also need principal curvatures for distance estimation purposes. These are the eigenvalues of the Weingarten map. Since matrices of the first fundamental form \( g \) and of the second fundamental form \( L \) are known, we can calculate some or all of these by solving the sparse generalized eigenvalue problem

\[
g v = \kappa_N(v) L v.
\tag{18}
\]

This yields the corresponding principal curvature direction \( v \).

### III. LOCAL INVERSION

Given a curve \( p \) in power space, and a solution \( p(0) = F(v(0)) \) of the power flow equations for its starting point \( p(0) \), one would often like to know the particular connected component \( v \) of the preimage of \( p \) in voltage space that contains \( v(0) \). Standard continuation methods become unstable near the SSB, where the Jacobian is nearly singular, because conceptually, they require multiplication of the inverse Jacobian by \( p'(t) \) to obtain \( v'(t) \).

To solve this problem, we have devised two alternative approaches [19]. Both are based on the idea of splitting the representation of the point \( v(t) \) into a pair consisting of a point \( q(t) \) on the SSB and a distance \( d(t) \) from \( q(t) \) in a particular direction \( w(q(t)) \), so that \( v = q + d w(q) \). Depending on the approach, the direction \( w(q) \) is chosen to be either the kernel of the differential at \( q \), or the surface normal \( N_v(q) \) (See (1) for a sketch). We do not elaborate on how to obtain the initial values of \( w \) and \( q \); for \( w = N_v \) one can use orthogonal projection ((V)), whereas for \( w = k \) we have so far used Newton’s method to obtain a solution, or gradient methods in case some point \( q^* \) on SSB close to \( v \) was known. In this case we used a gradient descent method to trace a curve on the SSB starting at \( q^* \) that would end up in a location \( q \) where the kernel would be collinear with the segment joining \( q \) and \( v \).

Both approaches allow us to stably compute \( v'(t) \) from \( p'(t) \) even in the vicinity of the SSB. Because the terms of \( F \) are at most quadratic, its Taylor series contains at most three terms and can be used to express the dependency of \( v'(t) \) on \( p'(t) \) by a well-conditioned linear equation system.

Both approaches should be combined: The kernel is easier to compute, needing fewer derivatives and usually providing more accuracy, but may become tangential to the SSB. If it becomes tangential, it is unreliable for the task of representing...
We know that the length of the vector \( p - F(q) \) scales exactly quadratic with \( d \) \((21)\), providing an alternative means to update \( d \). This exploitation of the quadratic folding behavior of \( F \) presented in the geometric splitting construction restricting the singular part of the map to the family of 1-d kernel vector lines each mapped to quadratically scaled Euclidean rays, lead here to geometric natural unfolding construction providing a high precision inversion.

According to Eq. (10), \( k^T (H F(q)) = D F(k) \) for all \( q \). This simplifies the formulas used here in several places.

We use this equation system to implement a continuation method that traces how the preimage \( v = q + d \cdot k \) of \( p \) evolves as \( p \) changes in the direction \( \dot{p} \). There are some problems with this that need to be addressed: First, as the curve \( q \) is constructed, it may deviate from the SSB due to numerical inaccuracies. If that happens, it should be corrected by projecting it back onto the SSB along the direction \( w(q) \). Second, we found in experiments that the simplification employed in Eq. (21), while valid when using exact arithmetic, leads to numerical errors that can be dramatically reduced in exchange for the rather small effort of using the full Eq. (20). Third, it may not always be possible or reliable to represent \( v \) as \( q + d \cdot k(q) \), namely if the kernel is (nearly) tangential. In that case, it would be more appropriate to use the normal instead of the kernel for \( w \). A suitable point \( q \) so that \( v \) can be expressed as \( q + d \cdot N_v(q) \) can be found using orthogonal projection (see (V)). Using the normal should be avoided when the kernel is available for representing \( v \) because the differential of the normal is much more expensive to compute.

For these reasons, we should also derive the equation system for \( \dot{q} \) and \( \dot{k} \) when the full (20) is used. Differentiating Eq. (20) with respect to \( t \), we get:

\[
\dot{p} = (D F) \dot{q} + d \cdot (D F) w + d \cdot \dot{q}^T (H F) w + d^2 \cdot \dot{k}^T (H F) k.
\] \((24)\)

To handle the case \( w := k \), we can as above derive a linear equation system from this, solving which tells us \( \dot{q} \), \( \dot{k} \) and \( \dot{d} \), given \( \dot{p} \). For the case \( w := N_v \) we use the directional derivative of \( N_v(q) \) in the direction \( \dot{q} \), which is given by \((D N_v(q)) \dot{q}\). For this, we have so far used and tested our methods in II for explicitly computing the Weingarten matrix.

We now consider an improved geometric method with much lower complexity, which still needs to be tested.)

A. Numerical Results

We tested the local inversion algorithm on power network configurations from the power system test case archive of the University of Washington \[10\]. In particular, we compared the precision of three of our methods: The method that uses the kernel and omits the linear term of the Taylor expansion that is theoretically zero, the method that uses the kernel but includes the linear term, and the method using the normal vector. We start with curves in voltage space, map them through \( F \) into power space and invert the results back into voltage space using the three algorithms described above. By comparing the
result with the original curve, we can estimate the precision, which is shown in (I) (taken from [19]).

We point out that all these test implementations as well as all other ones were very far away from being optimized. They were rather first experimental proofs of concept that all those computations are possible.

Using the linear term in the kernel method pays off in precision, for almost no additional runtime cost.

A comparison of execution times for a single step is displayed in (II).

IV. JACOBEAN HESSIAN OF LOCAL INVERSE

In III we presented well-tested methods tracing for given curves in power space close to and on the SSB local inverse (pre-image) curves in voltage space by computing its tangent vectors. This gave us all first-order directional derivatives of the local inverse $F^{-1}$ in the respective local set in power space, thus especially all partial derivatives hence the Jacobean of that local inverse $F^{-1}$.

In [16] formula (67) we presented a method (that we have tested) for directly computing the second-order derivative $\ddot{v}(t)$ of a preimage curve $v(t) = F^{-1}(p(t))$. One could also compute second-order derivatives of preimage curves $v(t)$ of given curves

$$p(t) = F(v(t))$$

using differentiating (25) twice with respect to $t$ yields

$$\ddot{p}(t) = \dot{v}^T H(v(t))\dot{v}(t) + DF(v(t))\ddot{v}(t)$$

thus

$$DF(v(t))\ddot{v}(t) = \dot{v}^T H(v(t))\dot{v}(t) - \ddot{p}(t)$$

We avoided this equation (26) in case $D(F(v(t)))$ is ill conditioned. But having a decent inverse $D(F(v(t)))^{-1}$ one could use this equation without the need of computing and using the full mostly non-sparse inverse of the Jacobean. In our case with the Jacobean $D(F(v(t)))^{-1}$ available from III we yet preferred the way described in [16] formula (67) as to obtain $\hat{v}$ from $\hat{p}$ as we often only need second derivatives of selected image curves.

Since otherwise in this situation with $DF$ in (26) being ill conditioned we might alternatively multiply equation (26) from the left with $DF^{-1}$ as to get an equation for $\dot{\hat{v}}$. Even if that would work we would need computing the full non-sparse inverse $DF^{-1}$ which we wanted to avoid. Hence we preferred the direct method in [16] formula (67) for second order derivatives which also seems to be more accurate as well.\footnote{Clearly one also could approximate less accurately the second derivative by difference methods from the first derivatives.}

We proceed to compute the Hessian $HF^{-1}(p_0)$. For computing the Hessian $HF^{-1}(p_0)$ we already have the diagonal second-order partials $\partial^2_{s,s} F^{-1}(p_0)$ as we know how to compute second-order derivatives of pre-image curves and need the mixed partials $\partial_s \partial_p F^{-1}(v_0)$. For this, we define a two-variable embedding map $\varphi(s,t) = (sp_i + tp_j)$ in power space with basis vectors $p_i, p_j$, next we define $\psi(s,t) := F^{-1}(\varphi(s,t))$ and three curves $v_1(s) := \psi(s,0), v_2(t) := \psi(0,t)$ and $v_3(r) := \psi(r,r)$. The second-order derivatives of these three curves which we can compute yield the $(2,2)$ Hessian $H\psi(s,t)$ with four vectorial elements, where two diagonal ones are the second derivatives of $v_1, v_2$. Knowing those the second derivative of $v_3(r)$ yields the non-diagonal element using here that the second derivative of $v_3(r)$ fulfills the quadratic form $(s,t)H\psi(s,t)^T$ with $s = t = 1$ thus giving access to the $\partial v_3, \partial v_3$ providing the wanted mixed partial $\partial_s \partial_p F^{-1}(p_0)$.

V. ORTHOGONAL PROJECTION

For some purposes, such as our local inversion technique when using the normal vector, it is required that, given a point $v$, we find a point $q$ on the SSB so that the normal at $q$ points along the direction $v−q$. Since the SSB is the implicit surface where the absolutely smallest eigenvalue $\lambda$ of the power flow differential $D F$ is 0, we can use the approach described in the following to find that orthogonal projection $q$. Also of interest are orthogonal projections of entire curves onto the SSB [20] (see (2)).

For this, we can use the subsequent fairly delicate method searching for a locally minimal distance projection $q$ of space point $v$. We also successfully tested simpler gradient descent methods tracing curves on the SSB starting in a point $q$, near to $v$ that would end up in a point $q$, such that $(q−v)$ would be collinear with the normal $N_q$ on the SSB, with SSB either in voltage space or in power space, the latter being the easier case.

A. An Algorithm for Computing Orthogonal Projections of Points onto the SSB

Let $\lambda : \mathbb{R}^n \rightarrow \mathbb{R}$ be a function whose zero set is the surface that we want to project onto, such as the SSB with $\lambda$ being the smallest eigenvalue of $DF$, and let $v$ be the point that we want to project. First, we evaluate $s = \lambda(v)$. Then, we compute a parameterized curve $r$ so that $r(s) = v$, $r(0) = q$ and for all $t$ between 0 and $s$, the normal to the iso-surface with $\lambda(r(t)) = t$ and the direction $r(t)−v$ are linearly dependent.
Table I

| # of buses | Kernel w/o lin. | Kernel with lin. | Normal |
|------------|-----------------|-----------------|--------|
| 14         | 2.3877483365 \times 10^{-9} | 2.3770686421 \times 10^{-9} | 2.1982954847 \times 10^{-9} |
| 30         | 7.05835279589 \times 10^{-9} | 7.05834973039 \times 10^{-9} | 7.97487174331 \times 10^{-9} |
| 57         | 1.64770875908 \times 10^{-7} | 1.6477075383 \times 10^{-7} | 3.462434738 \times 10^{-9} |
| 118        | 6.71182537764 \times 10^{-8} | 1.07999471555 \times 10^{-7} | 1.9659196324 \times 10^{-9} |

Table II

| # of buses | Kernel w/o linear | Kernel with linear | Normal |
|------------|-------------------|-------------------|--------|
| 14         | 2.31940697575 \times 10^{-9} | 2.38882856666 \times 10^{-9} | 2.1966634049 \times 10^{-9} |
| 30         | 7.0583552083 \times 10^{-6} | 7.05835718598 \times 10^{-6} | 7.97486759134 \times 10^{-9} |
| 57         | 1.64760675278 \times 10^{-4} | 1.64772007711 \times 10^{-4} | 3.4938430007 \times 10^{-4} |
| 118        | 5.37431102992 \times 10^{-7} | 1.1515965505 \times 10^{-5} | 2.043978252 \times 10^{-7} |

Runtime for a single step of the algorithm in seconds on a laptop with an Intel i7 7700HQ processor.

See Fig. 3 for an illustration of the idea. The condition for lying on the isosurface is equivalent to the existence of a kernel \( k \) of \( DF - \lambda I \). We make the generic assumption that the gradient of \( \lambda \) does not vanish\(^2\), so that \( r(t) - v \) can always be written as a multiple of it:

\[
d(t) \cdot (D \lambda(r(t)))^\top = v - r(t).
\] (27)

The expressions \( D\lambda \) (and its directional derivatives needed below) may be calculated analogously to (3). Then, we trace the curve \( r \) from \( t = s \) to \( t = 0 \) while maintaining this relation to stay true. For this, we differentiate with respect to \( t \) (denoted by a dot on top of symbols) and add the condition \( \dot{\lambda} = (D\lambda(r))\dot{r} = \text{sgn } s \):

\[
\begin{pmatrix}
d \cdot (H\lambda(r)) + 1 & (D\lambda(r))^\top \\
0 & 0
\end{pmatrix}
\begin{pmatrix}
\dot{r} \\
\dot{s}
\end{pmatrix} =
\begin{pmatrix}
0 \\
\text{sgn } s
\end{pmatrix}.
\] (28)

We integrate this ODE system from \( t = s \) to \( t = 0 \) and obtain the orthogonal projection \( q = r(0) \) onto \( \lambda = 0 \) surface.

There is, however, one problem with this: The matrix becomes singular as a focal surface of the \( \lambda = t \) surface approaches \( q \). If the ODE solver does not adapt its step size, this leads to inaccurate results, whereas if it does adapt, the step size may approach zero and the solver gets stuck or an attempt to invert a singular matrix is made. The solution is to limit the step size from below and use Eq. (28) as the predictor in a kind of predictor-corrector algorithm. We found that a good choice for the corrector step is to replace \( r(t) \) with \( \text{arg min}_{\lambda(r)\epsilon t=\text{sgn } s ([\tilde{r} - v])} \), found using projected gradient descent initialized with \( r(t) \). Afterwards, \( \dot{d} \) needs to be updated as well: \( d := \tilde{d} = \frac{D\lambda(r)(v-q)}{\| D\lambda(r)(v-q) \|^2} \). With a high-order adaptive step size control scheme (We used Dormand-Prince), the corrector usually has nothing to do because the predicted point already is very accurate. Only when the step size limitation becomes relevant does it have to do a few iterations.

B. Tracing the Orthogonal Projection of a Curve

Once we have found the orthogonal projection of a point \( v \), we can compute its derivatives with respect to changes in \( v \), and hence we can trace the orthogonal projection of a differentiable curve. So, let \( v \) and \( q \) now again be curves parameterized by curve parameter \( t \) (unrelated to the use of \( t \) in the previous paragraph), and let \( d \) be a real-valued function of \( t \) so that \( v = q + d N_V(q) \), where \( N_V(q) \) is the unit normal to the SSB in voltage space. \(^3\) Then \( d = (v - q)^\top N_V \), and by differentiating these with respect to \( t \), we get

\[
\dot{v} - \dot{q} = \dot{d} \cdot N_V(q) + d \cdot (D N_V)\dot{q}
\] (29)

\[
\dot{d} = (v - q)^\top N_V(q) + (v - q)^\top (D N_V(q))\dot{q}.
\] (30)

\(^3\)We tested the respective tracing method also for the SSB in power space cf. [17], [19]; here we report details only on the voltage space case.
Here, $\dot{q}$ is orthogonal to $N(q)$ and $(v - q)$ is parallel to $N_v(q)$ and thus orthogonal to the image of $D N_v(q)$, which is the tangent plane:

$$\dot{d} = \dot{v}^T N_v(q),$$

(31)

hence (29) becomes the following linear equation system for $\dot{q}$, given $\dot{v}$:

$$(d \cdot (D N_v(q)) + 1)\dot{q} = \dot{v} - (\dot{v}^T N_v(q)) N_v(q).$$

(32)

(8) tells how $D N = -W$ can be computed.

The function $d$ tells the distance between $v$ and $q$. When it is smaller than the smallest positive\(^4\) radius of curvature of the SSB at $q$, $q$ is a point on the SSB guaranteed to be locally closest to $v$. The smallest positive radius of curvature is the reciprocal of the largest eigenvalue of the Weingarten map. If $d$ is larger than the smallest radius of curvature, then $q$ cannot possibly be the closest point to $v$ on the SSB. See (4) for an illustration, adapted from the master thesis of Gruhl [17].

C. Local topology of SSB

We focus on discussing the local SSB topology in voltage space. Often computations are implicitly restricted to analyzing restrictions of the power flow map $F : V \rightarrow P$ practically to a local neighborhood $U(v_0)$ of a point $v_0$ on the SSB in voltage space, assuming that $U(v_0)$ is a topological unit half disc with boundary on the SSB, thus topologically equivalent to the bordered half-space of $\mathbb{R}^n$ with euclidean norm $|v| < 1$ and coordinate $v^\alpha$ of $v$ being positive.

Indeed for practical computing mostly only solutions of the power flow map to its restriction being the homeomorphism

$$F : U(v_0) \rightarrow F(U(v_0))$$

are relevant. Thus like in many other power flow papers this is a silent technical assumption we implicitly generally made in this paper as well as we did not want to mess with multiple possibly irrelevant solutions, excluding them by restriction to case (33). Without this assumption, the need for formal pre-image specification would aggravate precise descriptions of computations without improving insights.

\(^4\)“Positive” means curving towards $v$; We explain here only the case where $d \leq 0$ and therefore the normal points away from $v$.

In [16] we proved a practical criterion assuring that the restriction $F$ in (33) would be locally possible as it could be tested that locally close to some point on the SSB, the SSB would be a proper regular rank $n - 1$ differential hyper surface iff at $v_0$ the gradient of $\det(DF(v_0))$ is nonzero, the latter being equivalent to the easy test that the kernel (zero space) of $DF(v_0)$ has dimension one. Beyond that by inspecting low dimensional situations the generically $n - 1$ dimensional hyper surface structure of the SSB even in dimension $3$ easily may have partial $n - 2$ dimensional submanifolds, cf. fig. 5.

Beyond the practical relevance for checking necessary conditions for at least locally topologically safe conditions for numerically computing solutions for power flow equations we finally would like pointing to an also practically relevant insight disproving a concrete erroneous result in a major practically important report [21]. The latter erroneous result got by [21] highly precise numerical spectral computations was observing fork bifurcations in planar intersection curves of SSB in voltage space contained in the intersection set of SSB with a planar family of voltage rays. Those fork type topological bifurcations observed in [21] cannot occur as the topological structure of the zero set of $\det(F(v))$ here restricted on a 2d-plane being a real zero set of an algebraic function in two real variables can only have points with an even number of out going branches, cf. [16], [21].

D. Geodesic coordinates for solution sets

We consider the potential advantage of systematic and comprehensive numerical parameterizations using geodesic coordinates for all types of constrained solution sets of power flow equations being likely the most important advantage for using computational differential geometry in power flow computing. For detailed descriptions we refer to [16], and present here only figures thereby briefly indicating options for using respective numerical geodetic coordinates. Connected components of solution sets of nonlinear equation systems
underlying non-linear algebraic constraints even if they gener-
cally define regular submanifolds or only 2d-surfaces in high-
dimensional Euclidean space are generally hard to endow
with coordinates. We view these coordinates as numerical land
charts supporting systematic efforts for registering and accessing
in principle all solution points in the connected component
of the submanifold partial to the algebraic solution set. Here
our computational engineering seemingly still new concept
employing numerical geodetic coordinates generally appears
being the only generic way providing systematic approaches
towards the aforementioned goals. This is so partly the validity
of the theorem of Hopf and Rinow: Saying in the situation
that any point in the connected component of the solution set
can be reached from any other point in the component by a (shortest) geodesic line contained in the component if that
geodesic starts with the right initial direction. After fixing the
initial direction the 1d-path of the geodesic is uniquely defined.
This yields in principle a method for systematically reaching
and covering all points in the connected component of the
solution set and thus comprehensively and systematically
parameterizing all the component of the solution set. Even
more always locally in a neighborhood of the start point
and under some conditions globally this parametrization, eg.,
by geodesic polar coordinates is a diffeomorphism. Beyond
that we can compute and use the Jacobian of those geodesic
coordinates to trace a geodesic connecting path back into the
euclidean parametrising coordinate space.

In our power flow setting, we have the advantage that
the solutions set of the involved algebraic equations (subject
to constraints) typically relate to sparse quadratic equations.
This makes it computationally feasible computing families of
geodesics contained in and used to parametrize complicated
high dimensional submanifolds to the extent being needed
despite the constraints that may nastily constrain them to sub-
manifolds of the SSB. Here we only indicate the above concept
descriptively via figure 4 graphically illustrating a sample case
referring to a special situation in power flow computing. We
have here a four-dimensional power flow system say defined
by a map F from 4d voltage space to 4d power space, where
we have two passive and two active power coordinates, their
technical meaning being exchangeable in the context of our
computing. Now, in this case, the solution space boundary
is a 3d-hyper surface in both voltage and power space. In
the context at hand a 2d-manifold in the 3d-SSB has been
defined by the additional constraint that it must be contained
in a 3d-euclidean unit sphere in voltage space, thus points
with Euclidean norm \( |v| = 1 \); (it could have been any other
constraint as well. This one was graphically convenient as it
allowed projecting this unit sphere nicely stereographically
into euclidean 3d-Space.) Now we immediately numerically
can match between say the two active and the two passive
power parameters employing the underlying geodesic polar
coordinates inducing respective 2d- polar coordinates on the
(grey) 2d- manifold in voltage space as well as on the two
different respective 2d- planes in power space described by
\( (p_1, p_2) \) and \( (p_3, p_4) \) respectively. This would provide trans-
formations based on geodesic coordinates, i.e, answering in
this situation, (with all points partial to 2D-surface in the
3D-SSB) what would for (given \( p_1, p_2 \) parameters) be the
missing matching \((p_3, p_4)\) parameters, needed to complete the coordinates assuring the point in 4D would still comply with all given constraints. This prototype concept can be extended to far more complicated higher dimensional scenarios.

VI. CONCLUSION AND FURTHER WORK

We presented various formulas and algorithms for calculating geometric entities associated with the SSB, such as curvatures and orthogonal projections of points and curves. We laid special focus on the SSB in voltage space, which historically has received less attention and which is less accessible computationally. Our algorithm for local inversion of the power flow map allows us to continue a solution to the power flow equations along a curve with high precision. We omitted deeper results on the local topological structure of SSB, algorithms for computing geodesics and Jacobians of geodesic coordinates on SSB, and algorithms for improving the results of optimal power flow computing as to assure certain security constraints a posteriori. We plan to publish these results regularly, for now, they are accessible in [16].

Acknowledgements

This work had its origin during a research term of F.-E. Wolter at MIT in summer 2016, supported by a MISTI (MIT Germany) research grant that he initiated together with T. Sapsis, N. Patrikalakis and S. Karaman. This group was later joined by K. Turitsin. During this research term, F.-E. Wolter came into contact with electrical power researchers from Argonne Laboratory (D. Molzahn) and the University of Michigan (Ian Hiskens), who later invited him to visit [2]. All these scientists have been supporting the advancement of the ideas presented in this paper via various discussions and references, which had been both enlightening and encouraging. During visits to the IMI, NTU, which included seminars there [4], [5], work done in this project evolved further. Most recently, F.-E. Wolter, during his 2019 visit in Singapore [6], had the chance to come into contact with power flow researcher H. D. Nguyen who provided stimulating questions. We thank Hannes Thielhelm for many stimulating informal discussions with various researchers there, as well as of course for the generous invitations by N. Thalmann that helped building new promising cooperations in Singapore as well as supporting presenting the results to the computer graphics community [5].

REFERENCES

[1] F.-E. Wolter, P. Blanke, H. Thielhelm, and A. Vais, “Computational differential geometry contributions of the Welfenlab to GRK 615: modelling, simulation and software concepts for scientific-technological problems,” pp. 211–235, 2011.

[2] Wolter, F-E, “Computations in Riemannian space involving geodesics with their singularities and computing and visualizing singularities of dynamical systems using numerical methods based on differential geometry,” 2016, Michigan Power & Energy Laboratory Seminar and MIT Sandlab, http://sandlab.mit.edu/images/Seminar_series_spe Slides/Franzi_2016.pdf.

[3] Wolter, Franz-Erich, “Foundations for a differential geometric analysis of computations for powerflow and power grid,” 2017, MIT Seminar of Turitsin and Sapsis, https://drive.google.com/file/d/0BwvGPB1mxOrmdVfRHNHJYVFFcD9/view.

[4] Wolter, F-E, “Foundations for a differential geometric analysis of computations for power flow computing,” 6 2018, IMI research seminar, NTU, https://drive.google.com/file/d/11_UowvFjV99X89ezDmGXM3X0171Y9vGY/view.

[5] Wolter, F-E, “Foundations for a differential geometric analysis of computations for power flow computing,” 6 2018, CGI2018, NTU, http://www.cgs-network.org/cgi18/?page_id=1817.

[6] Wolter, F-E, “New geometric concepts for power flow analysis,” 3 2019, BTC Workshop, NTU.

[7] Y. Makarov, D. Hill, and I. Hiskens, “Properties of quadratic equations and their application to power system analysis,” International Journal of Electrical Power & Energy Systems, vol. 22, no. 5, pp. 313–323, 2000.

[8] I. Dobson, “The irrelevance of load dynamics for the loading margin to voltage collapse and its sensitivities,” in Bulk power system voltage phenomena III, voltage stability, security & control, Proceedings of the ECC/NSF workshop, Davos, Switzerland, 1994.

[9] I. A. Hiskens and R. J. Davy, “Exploring the power flow solution space boundary,” IEEE transactions on power systems, vol. 16, no. 3, pp. 389–395, 2001.

[10] R. Christie, “Archive of Testcases of the University of Washington,” https://www2.ee.washington.edu/research/pstca/, 2017.

[11] F. Alvarado, I. Dobson, and Y. Hu, “Computation of closest bifurcations in power systems,” IEEE Transactions on Power Systems, vol. 9, no. 2, pp. 918–928, 1994.

[12] I. Dobson, “Computing a closest bifurcation instability in multidimensional parameter space,” Journal of nonlinear science, vol. 3, no. 1, pp. 307–327, 1993.

[13] I. Dobson and L. Lu, “New methods for computing a closest saddle node bifurcation and worst case load power margin for voltage collapse,” IEEE Transactions on Power Systems, vol. 8, no. 3, pp. 905–913, 1993.

[14] I. Dobson, “Distance to bifurcation in multidimensional parameter space: Margin sensitivity and closest bifurcations,” Bifurcation control, pp. 704–706, 2003.

[15] M. Yao, I. A. Hiskens, and J. L. Mathieu, “Improving power system voltage stability by using demand response to maximize the distance to the closest saddle-node bifurcation,” 2018 IEEE Conference on Decision and Control (CDC), pp. 2390–2395, 2018.

[16] F.-E. Wolter and B. Berger, “Differential geometric foundations for power flow computations,” arXiv preprint arXiv:1903.11111, 2019.

[17] N. Gruhl, “Foundations for a differential geometric analysis of computations for power flow and power grid,” Master’s thesis, Leibniz Universität Hannover, 2017, https://drive.google.com/file/d/0BwvGPB1mxOrmdVfRHNHJYVFFcD9/view.

[18] F.-E. Wolter and S.-T. Tuohy, “Curvature computations for degenerate surface patches,” Computer Aided Geometric Design, vol. 4, no. 9, pp. 241–270, 0 1992.

[19] N. Hein, “Differentialgeometrische Analyse von Power Flow und Power Grid Berechnungen,” Master’s thesis, Leibniz Universität Hannover, 2017, https://drive.google.com/file/d/0BwvGPB1mxOrmdVfRHNHJYVFFcD9/view.

[20] J. Pegna and F.-E. Wolter, “Surface curve design by orthogonal projection of space curves onto free-form surfaces,” Transactions-American society of mechanical engineers journal of mechanical design, vol. 118, pp. 45–52, 1996.

[21] E. Ghys, “A singular mathematical promenade,” arXiv preprint arXiv:1612.06373, 2016.

[22] Y. V. Makarov, B. Vyakaranam, Z. Hou, D. Wu, D. Meng, S. Wang, S. T. Elbert, L. E. Miller, and Z. Huang, “Non-iterative voltage stability,” Pacific Northwest National Laboratory (PNNL), Richland, W A (United States), Tech. Rep., 2014.

[23] H. Thielhelm, A. Vais, and F.-E. Wolter, “Geodesic bifurcation on smooth surfaces,” The Visual Computer, vol. 31, no. 2, pp. 187–204, 2015.

[24] M. Gutschke, A. Vais, and F.-E. Wolter, “Differential geometric methods for examining the dynamics of slow-fast vector fields,” The Visual Computer, vol. 31, no. 2, pp. 169–186, 2015.