Robustness to Algorithmic Singularities and Sensitivity in Computational Kinematics

Luis Gracia*,[a], Jorge Angeles[b]

[a]*Instituto IDF, Universidad Politécnica de Valencia, Camino de Vera s/n, 46022 Valencia, Spain*

[b]*Department of Mechanical Engineering, McGill University, 817 Sherbrooke St., Montreal H3A 2K6, Canada*

Abstract

A robust approach to computational kinematics intended to cope with algorithmic singularities is introduced in this paper. The approach is based on the reduction of the original system of equations to a subsystem of bivariate-equations, as opposed to the multivariate-polynomial reduction leading to the characteristic univariate-polynomial. The effectiveness of the approach is illustrated for the exact function-generation synthesis of planar, spherical, and spatial four-bar linkages. Some numerical examples are provided for the case of the spherical four-bar function generator with six precision points to show the benefits of the proposed method with respect to methods reported in the literature.

Key words: Computational kinematics, Function-generation, Planar, spherical and spatial four-bar linkages

*Corresponding author.

Email addresses: luigraca@isa.upv.es (Luis Gracia), angeles@cim.mcgill.ca (Jorge Angeles)

Preprint submitted to IMechE, Part C *August 30, 2010*

1. Introduction

The kinematics of mechanical systems such as mechanisms and robots leads to a set of equations that involve a number of joint variables. These variables, e.g., joint angles, usually appear in these equations in a nonlinear form through their harmonic functions. The set of underlying kinematics equations must be solved for either analysis or synthesis purposes. In this regard, some transformations, e.g. the well-known *tan-half identities*, are commonly applied to obtain one single univariate nth-degree polynomial equation. The n roots of this polynomial, which is referred to as the *characteristic polynomial* of the system at hand, represent the solution sought. Raghavan and Roth [1] as well as Lee et al. [2] solved the inverse displacement problem of general six-revolute (6R) manipulators leading to a univa riate 16th-degree polynomial equation. Innocenti and Parenti-Castelli [3] and Gosselin et al. [4] obtained a univariate eighth-degree characteristic polynomial for the direct displacement problem of spherical parallel robots. Moreover, Alizade and Kilit [5] and Cervantes-Sánchez et al. [6] synthesized the four-bar spherical function generator (SFG) with five precision points based on a cubic polynomial. Similarly, Dukkipati [7] and Cervantes-Sánchez et al. [6] designed the SFG with six precision points via a 16th-degree and a 10th-degree polynomial, respectively.

However, the classical univariate-polynomial approach for computational kinematics has significant drawbacks, such as algorithmic singularities and ill-conditioning, which are illustrated in Section 2 with several examples. In order to overcome these drawbacks, which are *algorithmic*, rather than intrinsically kinematic or structural, the authors propose an alternative means, the bivariate-equation approach, to solve robustly problems in computational kinematics of the above kinds. While the underlying idea has been proposed in various publications by the second author and his team [8, 9], the generality of the bivariate approach and its comparison with the classical alternative have not as yet been reported. The procedure of the proposed approach is described in Section 3, where it is illustrated for the exact function-generation synthesis of planar, spherical, and spatial four-bar linkages. Some numerical examples are discussed in Section 4 for the SFG to show the benefits of the proposed approach vis-a-vis its counterpart alternatives. Finally, Section 5 includes the conclusions.

2. Singularities and Sensitivity in the Classical Univariate-polynomial Approach

2.1. Polynomial Root-finding Sensitivity

Although there exist methods in the literature to obtain the numerical solution of polynomials [10], the problem of polynomial root-finding has been identified as ill-conditioned by numerical analysts for some time [11]. Ill-conditioning here stems from the nature of the problem itself, rather than because of the intrinsically finite accuracy of floating-point arithmetic. Ill-conditioning implies high sensitivity to round-off errors, e.g., high error amplification in polynomial root-finding, in the sense that small perturbations in the data lead to dramatic changes in the root values. That is, when polynomial coefficients are slightly perturbed, the computed root values become corrupted with unacceptably large error. As a consequence, polynomial root-finding should be avoided in any type of computation, whenever possible. To illustrate the sensitivity in question, let us consider the 20th-degree polynomial proposed by Forsythe [11]:

$$P(x) = (x-1)(x-2)\cdots(x-19)(x-20) = x^{20} - 210x^{19} + \cdots + 20! \quad (1)$$

The roots of $P(x)$, $\{1, 2, ..., 19, 20\}$, are well separated. Suppose that a change in the tenth most significant digit is made in *only one* of the twenty coefficients. For example, if the coefficient of x^{19} is changed from -210 to $-210 - 10^{-7}$, the computed roots for the modified polynomial, i.e. $\tilde{P}(x) = P(x) - 10^{-7}x^{19}$, are: 1, 2, 3, 4, 5, 6, 6.99, 8.01, 8.92, $10.12 \pm 0.60\imath$, $11.82 \pm 1.59\imath$, $14.01 \pm 2.44\imath$, $16.72 \pm 2.73\imath$, $19.45 \pm 1.87\imath$, 20.78. Note that a negligibly small change in one coefficient has caused 10 of the roots to become complex, two of them having moved more than 2.73 units off the real axis. The reason why these roots changed so dramatically is not a round-off problem; it is a matter of intrinsic problem sensitivity.

2.2. Algorithmic Singularities

This kind of singularities arises when the tan-half identities are introduced to obtain the characteristic univariate-polynomial of a specific kinematics problem. Moreover, in function-generation synthesis the linkage geometric parameters are transformed into dimensionless design parameters via nonlinear transformations, e.g., those introduced by Freudenstein [12]. These transformations give rise to singularities that are algorithm-dependent, rather than intrinsic to the mechanical system at hand. For example, when the tan-half-angle identities are used, an algorithmic singularity is apparently introduced at the angle value of π. The outcome is that, if one of the roots finds itself at that value, then the characteristic polynomial admits one solution at infinity, which manifests itself as polynomial *deflation*. This phenomenon is illustrated with one example.

Serial decoupled 6R robots are those whose last three revolute axes intersect at a common point, which is known as center of the wrist. For this type of robots, the inverse displacement problem for the first three joints—i.e., the positioning problem of the wrist center—can be reduced to two bivariate trigonometric equations [8]:

$$h_1 c\theta_1 + h_2 s\theta_1 + h_3 c\theta_3 + h_4 s\theta_3 + h_5 = 0, \tag{2}$$

$$h_6 c\theta_1 + h_7 s\theta_1 + h_8 c\theta_3 + h_9 s\theta_3 + h_{10} = 0, \tag{3}$$

where $\{h_i\}_1^{10}$ are constant coefficients computed from the data, i.e. the desired position of the wrist center and the robot Denavit-Hartenberg parameters. In the above equations a compact trigonometric notation has been used: $c\theta_i \equiv \cos(\theta_i)$, $s\theta_i \equiv \sin(\theta_i)$, where θ_1 and θ_3 are the angles of the first and the third joints, respectively.

Upon introduction of the tan-half identities and then elimination of θ_1 from the equations thus resulting by dialytic elimination [13], the above equations can be reduced to a single quartic univariate polynomial equation:

$$h_{11}\tau_3^4 + h_{12}\tau_3^3 + h_{13}\tau_3^2 + h_{14}\tau_3 + h_{15} = 0, \quad \tau_3 = \tan(\theta_3/2), \tag{4}$$

which is the characteristic polynomial at hand. The above equation in τ_3 of the given problem admits four roots. Thus, up to four possible values of θ_3 are expected. Once each value of θ_3 is available, the *unique* value of θ_1 is computed from Eqs. (2) and (3). Then, the value of the second joint angle, θ_2, can be readily computed from a set of linear equations [8], not given here for the sake of compactness.

For the isotropic 3R robot in Fig. 1a, with the wrist center at $(0, a, 0)$, where a is a non-zero length, Eqs. (2) and (3) become:

$$\mathcal{C}_1: \quad s\theta_1 + c\theta_3 + s\theta_3 + 1 = 0, \tag{5}$$

$$\mathcal{C}_2: \quad c\theta_1 + s\theta_3 + 1 = 0. \tag{6}$$

For this case, the quartic Eq. (4) deflates into a cubic equation:

$$\tau_3^3 + 2\tau_3^2 + 3\tau_3 + 1 = 0, \tag{7}$$

whose roots are readily found as -0.43 and $-0.785 \pm 1.307\imath$. That is, only one real solution is obtained: $\theta_3 = 5.47$ rad. However, the configuration shown in Fig. 1a is a quite symmetric posture of this manipulator, which does not correspond to the real solution obtained above. In fact, the solution yielding the posture in Fig. 1a disappeared because of the use of the tan-half-angle identities. The two contours derived from Eqs. (5) and (6) are displayed in Fig. 2, their intersections yielding the two real roots, including the one leading to the posture in Fig. 1a. The missing root is, hence, $\tau_3 \to \pm\infty$, whence, $\theta_3 = \pi$. For each value of θ_3, angle θ_1 is readily calculated from Eqs. (5) and (6) as 4.43 and π, respectively.

2.3. Conditioning of the Solutions

For the same isotropic 3R robot with the wrist center at $(0, 2a, -a)$, as shown in Fig. 1b, Eqs. (2)–(4) become:

$$\mathcal{C}_1: \quad 2s\theta_1 + c\theta_3 + s\theta_3 - 1 = 0, \tag{8}$$

$$\mathcal{C}_2: \quad 2c\theta_1 + s\theta_3 + 1 = 0, \tag{9}$$

$$\tau_3^4 - 4\tau_3^3 + 2\tau_3^2 + 4\tau_3 - 3 = 0. \tag{10}$$

The foregoing quartic equation in τ_3 has four real roots, namely, 1, 1, -1, and 3, yielding four solutions with the joint angle values:

$$R_1(\pi, \pi/2), \qquad R_2(\pi, \pi/2), \qquad R_3(\pi/2, -\pi/2), \qquad R_4(2.5, 2.5). \tag{11}$$

The two contours derived from Eqs. (8) and (9) are displayed in Fig. 3a, their intersections yielding the four points R_i.

Now, let us consider a 10% perturbation in only one coefficient of Eqs. (8) and (9). That is, a value of 0.1 is added to and subtracted from Eq. (8):

$$\mathcal{C}_{1+}: \quad 2s\theta_1 + c\theta_3 + s\theta_3 - 0.9 = 0, \tag{12}$$

$$\mathcal{C}_{1-}: \quad 2s\theta_1 + c\theta_3 + s\theta_3 - 1.1 = 0, \tag{13}$$

where subscripts + and $-$ are used to identify the sign of the perturbation introduced. The new real solutions for the joint angles from Eqs. (9) and (12) or (13) are:

$$\begin{array}{ll} R_{1+}(\pi + 0.138, \pi/2 - 0.195), & R_{2+}(\pi + 0.029, \pi/2 + 0.041), \\ R_{3+}(\pi/2 + 0.003, -\pi/2 - 0.106), & R_{4+}(2.5 - 0.172, 2.5 + 0.258), \end{array} \tag{14}$$

$$\begin{array}{ll} R_{1-}(\text{No real solution}), & R_{2-}(\pi - 0.030, \pi/2 - 0.042), \\ R_{3-}(\pi/2 + 0.002, -\pi/2 + 0.096), & R_{4-}(\text{No real solution}). \end{array} \tag{15}$$

Thus, the percentage error amplification E (%) for each solution is:

$$E_{1+}(138, 195), \qquad E_{2+}(29, 41), \qquad E_{3+}(3, 106), \qquad E_{4+}(172, 258), \tag{16}$$

$$E_{1-}(\infty, \infty), \qquad E_{2-}(30, 42), \qquad E_{3-}(2, 96), \qquad E_{4-}(\infty, \infty), \tag{17}$$

where it is apparent that the perturbation has greater impact in solutions R_1 and R_4 than in solutions R_2 and R_3. In fact, the real solutions given by R_1 and R_4 are lost, as they become complex, for the negative 10% perturbation.

In order to find the cause of this behavior, the contours derived from Eqs. (8), (9), (12) and (13) are displayed in Fig. 3b. Note that contour $\mathcal{C}_1$ is almost tangent to contour $\mathcal{C}_2$ at roots R_1 and R_4, which gives high numerical sensitivity when perturbing the contours. In fact, the numerical conditioning of the solutions is given by the *angle* at which the contours intersect [8]. That is, for the best possible solutions, from the numerical conditioning viewpoint, the two contours cross each other at right angles, e.g., R_2 and R_3 in the above example. At ill-conditioned postures the contours are tangent to each other, e.g., R_1 and R_4 in the same example. A limit case of ill-conditioning can be found when the two contours overlap, thereby leading to a continuum of solutions. In this case purely numerical methods, like those used in the univariate-polynomial approach, are strongly recommended against, since they hide information available in the graph representation.

3. A Robust Method for Kinematic Computations

3.1. Overall Description

The proposed method, applicable to similar problems in computational kinematics, is based on a system of bivariate equations. In this approach, the equations of the kinematics problem at hand are reduced to a system of M (≥ 2) nonlinear equations in two unknowns. Note that only two such equations are required, although equation redundancy provides robustness to the method, as explained in Subsection 3.2. These equations are then displayed in the plane of the two unknowns using the graphics features of commercial software for computer algebra. The M contours must intersect at a solution. All the intersections are roughly estimated by *inspection* on the contour plots, and then *refined* by a numerical procedure to the desired accuracy. To this end, the M bivariate equations are regarded as independent, since possible contradictions, due to round-off error, may render them independent. Then, rather than one solution to all the bivariate equations, what the refinement algorithm seeks is their (nonlinear) least-square approximation, which can be done via the Gauss-Newton method [14], see Appendix B. To complete the solution, the remaining unknowns are computed for each pair of roots thus found.

3.2. Main Advantages

1. *Avoiding direct polynomial root-finding*: As mentioned in Subsection 2.1, polynomial root-finding is intrinsically an ill-conditioned problem and should be avoided whenever possible.
2. *Avoiding algorithmic singularities*: The proposed approach keeps the original nonlinear functions, i.e. the harmonic functions of the joint angles, in the bivariate equations, without any transformation. Thus, algorithmic singularities, e.g., polynomial deflation, as illustrated in Subsection 2.2, are avoided. Therefore, *all real* roots are found and high error amplification due to nearness to algorithmic singularities is prevented. This advantage is shown in Example 3 of Subsection 4.2
3. *Robustness to spurious roots and ill-conditioning*: Although only two bivariate-equations are independent, more than two are always welcome, as these add robustness to the computations involved: firstly, redundancy helps identify *spurious* roots, i.e., points where two or more, but not all, contours intersect; secondly, when two or more contours turn out to be either *tangent* or close to tangent at an intersection, the

remaining non-tangent contours help to pinpoint the intersection coordinates, as discussed in Subsection 2.3. These advantages are shown in Example 1 of Subsection 4.2.

4. *Obtaining approximate solutions*: The graph representation of contours is useful to notice when all of them *approximately* intersect or even overlap, which is very difficult to establish with purely numerical approaches. Then, approximate solutions or a continuum of approximate solutions, which are *almost* exact for the kinematics problem at hand, can be found. This advantage is shown in Example 2 of Subsection 4.2.

3.3. An Outline of the Robust Approach for Function-generation Synthesis

In this Subsection, a common procedure is described for the robust function-generation synthesis of four-bar linkages using the proposed bivariate-equation approach. In further Subsections, the procedure is applied to the planar, spherical, and spatial cases. The kinematic relations associated with the function-generation problem define the *input-output equation.* For the common approach, this equation is written in the form [12]:

$$\begin{aligned} p_1 + g_1(\psi, \phi, p_{N-1}, p_N) + p_2 g_2(\psi, \phi, p_{N-1}, p_N) \\ + \cdots + p_{N-2} g_{N-2}(\psi, \phi, p_{N-1}, p_N) = 0, \end{aligned} \tag{18}$$

where g_i is a generic nonlinear function, $\{p_i\}_1^N$ are the independent *design parameters* of the linkage, ψ is the *input* variable and ϕ is the *output* variable.

When used for the kinematic synthesis of the four-bar function generator, the input-output equation must be satisfied for N *precision points* (ψ_i, ϕ_i), thereby leading to N *synthesis equations*:

$$p_1 + g_{1i} + p_2 g_{2i} + \cdots + p_{N-2} g_{(N-2)i} = 0, \qquad \text{for} \quad i = 1, \ldots, N, \tag{19}$$

where $g_{ji} \equiv g_j(\psi_i, \phi_i \, p_{N-1}, p_N)$.

To eliminate parameter p_1 from the above equations, the Nth synthesis equation is subtracted from the remaining $N-1$ equations, to give:

$$g_{1i} - g_{1N} + p_2(g_{2i} - g_{2N}) + \cdots + p_{N-2}(g_{(N-2)i} - g_{(N-2)N}) = 0,$$
$$\text{for} \quad i = 1, \ldots, N-1. \tag{20}$$

The above system of $N-1$ equations can be expressed in array form:

$$\begin{bmatrix} \mathbf{A} & \boldsymbol{b} \end{bmatrix} \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix} = \bar{\mathbf{A}} \, \bar{\boldsymbol{p}} = \mathbf{0}, \tag{21}$$

where $\boldsymbol{p} = [\; p_2 \quad \cdots \quad p_{N-2} \;]^T$, $\mathbf{A}(p_{N-1}, p_N) \in \mathbb{R}^{(N-1)\times(N-3)}$, $\boldsymbol{b}(p_{N-1}, p_N) \in \mathbb{R}^{N-1}$, $\bar{\mathbf{A}}(p_{N-1}, p_N) \in \mathbb{R}^{(N-1)\times(N-2)}$, $\bar{\boldsymbol{p}} \in \mathbb{R}^{N-2}$ and $\mathbf{0} \in \mathbb{R}^{N-1}$.

Matrix $\bar{\mathbf{A}}$ must be rank-deficient (necessary condition) for Eq. (21) to admit a non-trivial solution. This means that the determinants of all of its $(N-2) \times (N-2)$ submatrices, obtained by deleting one of its $N-1$ rows, should vanish. Note that only two of these $N-1$ determinants are independent. However redundant determinants add robustness to the method, as discussed in Subsection 3.2. Therefore, the vanishing of $N-1$ determinants is imposed, which yields, correspondingly, $N-1$ contours in the p_{N-1}-p_N plane. The intersections of all contours then yield the (p_{N-1}, p_N) pairs of real values which render $\bar{\mathbf{A}}$ rank-deficient. For each intersection, the remaining design parameters in Eq. (21) are computed by linear-equation solving: given that the remaining system of linear equations is overdetermined, a least-square approximation is computed, which yields, in *symbolic* form:

$$\boldsymbol{p} = -\left(\mathbf{A}^T \mathbf{A}\right)^{-1} \mathbf{A}^T \boldsymbol{b}. \tag{22}$$

Finally, parameter p_1 can be computed from any of the N synthesis equations in Eq. (19).

3.4. Robust Function-generation Synthesis for the Planar Four-bar Linkage

The input-output equation of the planar four-bar linkage can be written in the form [12]:

$$k_1 + k_2 c\psi + k_3 c\phi - c\psi c\phi + s\psi s\phi = 0, \tag{23}$$

which is called the *Freudenstein equation*, where $\{k_i\}_1^3$ are the *Freudenstein parameters*, ψ is the input angle and ϕ is the output angle. Note that the above equation is linear in the Freudenstein parameters and bilinear in the harmonic functions of the input and output variables.

The case under study has three independent design parameters $\{k_i\}_1^3$. However, two additional design parameters can be introduced: the *reference angles* ψ_0 and ϕ_0, with respect to which the input and output angles are measured. If $\psi \to \psi + \psi_0$ and $\phi \to \phi + \phi_0$ are introduced into Eq. (23), an alternative input-output equation is obtained:

$$\begin{aligned} &k_1 + k_2 c(\psi + \psi_0) + k_3 c(\phi + \phi_0) \\ &\quad - c(\psi + \psi_0) c(\phi + \phi_0) + s(\psi + \psi_0) s(\phi + \phi_0) = 0. \end{aligned} \tag{24}$$

By comparing Eqs. (18) and (24), a set of equivalences is obtained: $N = 5$, $\{p_i\}_1^3 = \{k_i\}_1^3$ and $\{p_4, p_5\} = \{\psi_0, \phi_0\}$. Thus, the function-generation synthesis for this linkage is obtained as described in Subsection 3.3 with the foregoing equivalences.

3.5. Robust Function-generation Synthesis for the Spherical Four-bar Linkage

The input-output equation of the spherical four-bar linkage can be written in a form similar to that of the foregoing Freudenstein equation [12]:

$$k_1 + k_2 c\psi + k_3 c\phi - k_4 c\psi c\phi + s\psi s\phi = 0, \tag{25}$$

where $\{k_i\}_1^4$ are the four Freudenstein parameters, ψ is the input angle and ϕ is the output angle.

Again, two additional design parameters can be introduced: the reference angles ψ_0 and ϕ_0, with respect to which the input and output angles are measured. As above, if the transformations $\psi \to \psi + \psi_0$ and $\phi \to \phi + \phi_0$ are introduced into Eq. (25), an alternative input-output equation is obtained:

$$\begin{aligned} &k_1 + k_2 c(\psi + \psi_0) + k_3 c(\phi + \phi_0) \\ &\quad - k_4 c(\psi + \psi_0) c(\phi + \phi_0) + s(\psi + \psi_0) s(\phi + \phi_0) = 0. \end{aligned} \tag{26}$$

By comparing Eqs. (18) and (26), the equivalences are: $N = 6$, $\{p_i\}_1^4 = \{k_i\}_1^4$ and $\{p_5, p_6\} = \{\psi_0, \phi_0\}$. Thus, the function-generation synthesis for this linkage is obtained as described in Subsection 3.3 with the foregoing equivalences.

3.6. Robust Function-generation Synthesis for the Spatial RCCC Linkage

The spatial RCCC linkage is a single-input-double-output four-bar linkage whose input-output equations can be written in the form of the foregoing Freudenstein equations [15]:

$$k_1 + k_2 c\psi + k_3 c\phi - k_4 c\psi c\phi + s\psi s\phi = 0, \tag{27}$$

$$\begin{aligned} & k_5 + k_6 c\psi + k_7 c\phi - k_8 c\psi c\phi - k_2 s\psi + k_3 d s\phi \\ & \qquad + k_4(s\psi c\phi + d c\psi s\phi) + d s\psi c\phi + c\psi s\phi = 0, \end{aligned} \tag{28}$$

where $\{k_i\}_1^8$ are the eight Freudenstein parameters, ψ is the input angle, ϕ is the output angle and d is the output sliding. Note that the above equations are linear in the Freudenstein parameters and trilinear in the harmonic functions of the input and output angular variables plus the output sliding.

Four precision points (ψ_i, ϕ_i, d_i) can be selected using the eight Freudenstein design parameters. However, one additional precision point can be handled when two more design parameters are introduced: the reference angle ϕ_0 and the reference position d_0, with respect to which the output angle and the output sliding are measured. Therefore, if the transformations $\phi \rightarrow \phi + \phi_0$ and $d \rightarrow d + d_0$ are introduced into Eqs. (27) and (28), two alternative input-output equations are obtained:

$$k_1 + k_2 c\psi + k_3 c(\phi + \phi_0) - k_4 c\psi c(\phi + \phi_0) + s\psi s(\phi + \phi_0) = 0, \tag{29}$$

$$\begin{aligned} & k_5 + k_6 c\psi + k_7 c(\phi + \phi_0) - k_8 c\psi c(\phi + \phi_0) - k_2 s\psi \\ & \qquad + k_3 (d + d_0) s(\phi + \phi_0) + k_4 s\psi c(\phi + \phi_0) \\ & k_4 (d + d_0) c\psi s(\phi + \phi_0) + (d + d_0) s\psi c(\phi + \phi_0) + c\psi s(\phi + \phi_0) = 0, \end{aligned} \tag{30}$$

The 10 design parameters are computed in two stages: first, the five design parameters $\{k_i\}_1^4$ and ϕ_0 are obtained from Eq. (29); then, the remaining five design parameters $\{k_i\}_5^8$ and d_0 are obtained from Eq. (30). For the first stage, the equivalences with the common approach are $N = 5$, $\{p_i\}_1^3 = \{k_i\}_1^3$ and $\{p_4, p_5\} = \{k_4, \phi_0\}$. For the second stage, the equivalences with the common approach are $N = 5$, $\{p_i\}_1^3 = \{k_i\}_5^7$ and $\{p_4, p_5\} = \{k_8, d_0\}$.

3.7. Branch, Order and Circuit Defects in Function-generation Synthesis

These defects are inherent to linkage synthesis using precision points and may occur irrespective of the computational method used. Firstly, *branch defect* occurs when the synthesized linkage can pass through all the precision points only in different branch configurations, which would render the mechanism useless. For example, planar and spherical four-bar linkages present branch defect when the sign of the sine of the *transmission angle* changes [16]. Secondly, *order defect* occurs when the synthesized linkage fails to pass through the precision points in the specified order. Finally, *circuit defect* occurs when the input link has a *partial rotatability* with several disjointed ranges of motion and the precision points belong to different ranges.

Therefore, the linkages synthesized using the proposed approach should be *tested* for all three kinds of defect, then *discarded* if any defect is found. It is noteworthy that some works have been reported [16, 17] that propose means to avoid the above-mentioned defects. The *integration* of those techniques with the proposed approach could be investigated. However, this issue lies outside the scope of this paper, and is hence left aside, to be discussed in future work.

4. Examples for the Spherical Four-bar Function Generator

4.1. Kinematic Relations

Figure 4 depicts a general spherical four-bar linkage, which is characterized by four links connected by four revolute joints, namely, A, B, C and D, whose axes of rotation intersect at one common point O. Therefore, this point is the center of concentric spheres on which the motion of all the links takes place. In Fig. 4 the input and output angles are denoted by ψ and ϕ, respectively. Link dimensions are given by the angle between adjacent joint axes: α_1 for the input link, α_2 for the coupler link, α_3 for the output link and α_4 for the fixed link. The input-output equation of the spherical function generator (SFG) is known to be [18]:

$$\begin{aligned} f(\psi, \phi, \alpha_1, \alpha_2, \alpha_3, \alpha_4) &= (s\alpha_1 s\alpha_3)s\psi s\phi - (s\alpha_1 s\alpha_3 c\alpha_4)c\psi c\phi \\ &\quad + (s\alpha_1 c\alpha_3 s\alpha_4)c\psi + (c\alpha_1 s\alpha_3 s\alpha_4)c\phi + c\alpha_1 c\alpha_3 c\alpha_4 - c\alpha_2 = 0. \end{aligned} \tag{31}$$

Note that, when the input link length α_1 is equal to π or 2π the input angle ψ vanishes in the SFG input-output equation, which is equivalent to a zero length of the input link, where joint axes OA and OB collapse into one single axis. This situation corresponds to a degenerate design (structural singularity) of the SFG. An analogous situation occurs when either of the other link lengths is equal to π or 2π. Even several links can simultaneously degenerate. In any case, to be on the safe side regarding degenerate designs and mechanical constraints, the four link lengths must satisfy:

$$\alpha_i \in \;]0, \pi[\; \cup \;]\pi, 2\pi[, \qquad \text{for } i = 1 \ldots 4, \tag{32}$$

$$\alpha_i \le \alpha_j + \alpha_k + \alpha_m, \qquad \text{for } i = 1 \ldots 4, \text{ where } i \ne j \ne k \ne m. \tag{33}$$

Furthermore, a link dimension α_i close to 0, π or 2π is neither an acceptable design, as the linkage is close to a degeneracy and is, hence, fragile—as opposed to robust—i.e., highly sensitive to data perturbation.

The direct relations between the link lengths $\{\alpha_i\}_1^4$ and the Freudenstein parameters $\{k_i\}_1^4$, see Eqs. (25) and (31), are:

$$k_1 = (c\alpha_1 c\alpha_3 c\alpha_4 - c\alpha_2)/(s\alpha_1 s\alpha_3), \tag{34}$$

$$k_2 = (c\alpha_3 s\alpha_4)/s\alpha_3, \tag{35}$$

$$k_3 = (c\alpha_1 s\alpha_4)/s\alpha_1, \tag{36}$$

$$k_4 = c\alpha_4, \tag{37}$$

where the Freudenstein parameters $\{k_i\}_1^3$ are infinite only for degenerate designs. From the above expressions, the inverse relations between link lengths and Freudenstein parameters are:

$$\alpha_4 = \arccos(k_4), \tag{38}$$

$$\alpha_1 = \arctan(s\alpha_4/k_3), \tag{39}$$

$$\alpha_3 = \arctan(s\alpha_4/k_2), \tag{40}$$

$$\alpha_2 = \arccos(c\alpha_1 c\alpha_3 c\alpha_4 - k_1 s\alpha_1 s\alpha_3), \tag{41}$$

where the absolute values of parameter k_4 and of $(c\alpha_1 c\alpha_3 c\alpha_4 - k_1 s\alpha_1 s\alpha_3)$ must be smaller than or equal to 1 in order to obtain a real solution for the link lengths α_4 and α_2, respectively. With Eqs. (38)–(41) up to 64 ($= 2\times 4\times 4\times 2$) sets of $\{\alpha_i\}_1^4$ values for one single set $\{k_i\}_1^4$ are obtained, where 32 of these are the *antipodal* counterparts of the other 32. However, all these 64 sets give the same input-output equation and will, therefore, be considered in this work as the same design solution.

4.2. Numerical Examples[1]

In order to illustrate the synthesis of the SFG with six precision points, three examples of prescribed functions are considered: 1) the hyperbolic spiral, 2) the quadratic root and 3) the Archimedean spiral. The three functions

[1] The results reported here were obtained with Mathematica® 5.2.

and the corresponding set of precision points used are shown in Table 1. The first two are taken from [6], which is the only work that gives numerical examples for the problem at hand. However, the method reported in [6] introduces algorithmic singularities at the parameter values $\psi_0 = \pm\pi/2$ and $\phi_0 = \pm\pi/2$ due to the transformations made between design parameters and design coefficients.

Example 1: the hyperbolic spiral

For this example, the five contours in the ψ_0-ϕ_0 plane are displayed in Fig. 5. There are 20 intersections among all five contours, which are numbered from 1 to 5 because only five of them represent distinct designs. That is, for one intersection point in the range $[(0,0),(\pi,\pi)]$, there are other three symmetric intersection points obtained by subtracting π from the values of ψ_0 and/or ϕ_0. Therefore, to obtain all the possible solutions it suffices to analyze one $\pi \times \pi$ quadrant. For example, Fig. 6 displays the five contours in the range $[(-\pi/2,-\pi/2),(\pi/2,\pi/2)]$. According to Subsection 4.1, up to 256 ($= 4 \times 64$) sets of $\{\psi_0, \phi_0, \{\alpha_i\}_1^4\}$ values for the same design solution will be obtained, all of them giving the same input-output equation.

Although only two contours are independent, displaying all five contours adds *robustness* to the method, as discussed in Subsection 3.2. For example, in Fig. 6 three and four contours intersect at points $\{P_{3i}\}_1^5$ and $\{P_{4i}\}_1^3$, respectively, which can be readily identified as spurious roots because two and one contours, out of four, do not pass through these points. Moreover, at the first intersection point in Fig. 6, there are two sets of tangent contours; using all five contours ill-conditioning is avoided.

Table 2 shows the coordinates of each intersection point obtained with a nonlinear least-square minimization algorithm [14]. This table also shows the value of the *residual error* e_R and the value of the Freudenstein parameters for each point. Note that, for actual intersection points the value of the residual error tends to zero as the number of iterations of the algorithm is increased. The points that give a valid SFG design, which appear in boldface numbers, are those with $|k_4| \leqslant 1$. In particular, two valid designs are obtained, whose link-length values and reference angles can be found in Table 1. The plots in Figs. 7 and 8 illustrate for each design how the SFG accommodates its input-output function (*thin continuous line*) according to the precision points (*filled circles*) in order to satisfy the prescribed function (*thick continuous line*), regardless of its intricate shape. The two designs obtained here for this example are practically the same as the ones reported in [6].

Example 2: the square root

The five contours obtained for this example are displayed in Fig. 9. Mathematically, there are three independent intersection points among all of them. The value of the coordinates, residual error and Freudenstein parameters for the three points are shown in Table 3. Only the third point gives a valid SFG design, whose link-length values and reference angles can be found in Table 1. The plot in Fig. 10 illustrates how the designed SFG accommodates its input-output function to the precision points. This SFG design is practically the same as the one reported in [6].

However, let us further analyze the graph representation of the contours of this example. For this purpose, Fig. 11 displays the five contours in the range $[(0,0),(\pi,\pi)]$. Although mathematically the contours only intersect at three points, all of them almost *overlap* in the region $\mathrm{R_A}$, which lies approximately between the third intersection point and point 5_{A}.

To go deeper into this overlap, points 4_{A} and 5_{A} are treated as if they were actually intersection points, which they are not. Although the value of their residual error is not exactly zero, Table 3 shows that this error lies below 4×10^{-8}, which is quite acceptable. Moreover, point 4_{A} gives a valid design, whose link-length values and reference angles can be found in Table 1. In fact, the plot in Fig. 12 shows that this SFG design accommodates its input-output function to the precision points with no perceptible error. Therefore, in this example a continuum of *approximate* solutions has been found in the R_{A} region.

Example 3: the Archimedean spiral

The five contours are displayed for this example in Fig. 13. There are seven independent intersection points, as seen in Fig. 14. The value of the coordinates, residual error and Freudenstein parameters for the seven points are shown in Table 4. Only the first two points give a valid SFG design, whose link-length values and reference angles can be found in Table 1. Besides, the synthesis method developed in [6] was applied to the prescribed function and precision points of this example. The link-dimensions and reference angles obtained are shown in Table 1. In particular, two designs are obtained: the second is basically the same as its counterpart obtained with our method, whereas the first one is slightly different from its counterpart. The two designs of both approaches are comparatively depicted in Figs. 15 and 16. In particular, the first design obtained with the method developed in [6] exhibits a large error that is unacceptable for exact function-generation synthesis. The reason why this design fails lies in that it lies close to the algorithmic singularities—$\psi_0 = \pm\pi/2$ and $\phi_0 = \pm\pi/2$—introduced by the parameter transformations used in [6]. Thus, round-off errors are highly amplified, thereby giving rise to an unsuccessful design, as evidenced in Fig. 15.

5. Conclusions

In this paper a robust approach to computational kinematics was proposed. The approach is based on the reduction of the original system of equations to a subsystem of bivariate-equations, rather than the classical characteristic univariate-polynomial. The proposed method thus avoids the high sensitivity or error amplification that the univariate-polynomial approach exhibits at and around algorithmic singularities. The application of the approach was illustrated for the function-generation synthesis of planar, spherical, and spatial four-bar linkages. In particular, the numerical examples provided for the spherical four-bar function generator show the effectiveness of the proposed method with respect to methods reported in the literature.

It is noteworthy that the proposed approach can be used for solving *other* kinematics problems such as linkage synthesis for path generation or rigid body guidance, which remains as future work. In this sense, similar ideas to those proposed here have been successfully used for a robust analysis of the inverse kinematics of serial robots [8] and of the forward kinematics of spherical parallel robots [9].

Obviously, the proposed method is intended for *off-line* applications. However, it can be used in *real-time* problems—e.g. for finding real-time solutions of inverse kinematics problems—if the rough estimation of the contour intersections is *automated*. This can be done by means of a suitable *graphics system*, is another interesting field for future work.

Acknowledgments

The first author acknowledges the support of Universidad Politecnica de Valencia, research project PAID-00-09. The second author acknowledges the support of McGill University by means of a James McGill Professorship.

References

[1] **Raghavan, M.** and **Roth, B.** Inverse kinematics of the general 6R manipulator and related linkages. *ASME Journal of Mechanical Design*, 1993, **115**(3), 502–508.

[2] **Lee, H. Y., Woernle, C.,** and **Hiller, M.** A complete solution for the inverse kinematic problem of the general 6R robot manipulator. *ASME Journal of Mechanical Design*, 1991, **113**(4), 481–486.

[3] **Innocenti, C.** and **Parenti-Castelli, V.** Echelon form solution of direct kinematics for the general fully-parallel spherical wrist. *Mechanism and Machine Theory*, 1993, **28**(4), 553–561.

[4] **Gosselin, C. M., Sefrioui, J.,** and **Richard, M. J.** On the direct kinematics of spherical three-degree-of-freedom parallel manipulators of general architecture. *ASME Journal of Mechanical Design*, 1994, **116**(2), 594–598.

[5] **Alizade, R. I.** and **Kilit, O.** Analytical synthesis of function generating spherical four-bar mechanism for the five precision points. *Mechanism and Machine Theory*, 2005, **40**(7), 863–878.

[6] **Cervantes-Sánchez, J. J., Gracia, L., Rico-Martínez, J. M., Medellín-Castillo, H. I.,** and **González-Galván, E. J.** A novel and efficient kinematic synthesis approach of the spherical 4R function generator for five and six precision points. *Mechanism and Machine Theory*, 2009, **44**(11), 2020–2037.

[7] **Dukkipati, R. V.** *Spatial mechanisms: analysis and synthesis*, 2001 (CRC Press, Narosa Publishing House, Boca Raton, FL, USA).

[8] **Angeles, J.** *Fundamentals of robotic mechanical systems: theory, methods, and algorithms*, 2007 (third ed., Springer-Verlag, New York, NY, USA).

[9] **Bai, S., Hansen, M. R.,** and **Angeles, J.** A robust forward-displacement analysis of spherical parallel robots. *Mechanism and Machine Theory*, 2009, **44**(12), 2204–2216.

[10] **Sommese, A. J.** and **Wampler, C. W.** *The numerical solution of systems of polynomials arising in engineering and science*, 2005 (World Scientific Press, Singapore).

[11] **Forsythe, G. E.** Pitfalls in computation, or why a math book isnt enough. *The American Mathematical Monthly*, 1970, **77**(9), 931–956.

[12] **Freudenstein, F.** Approximate synthesis of four-bar linkages. *Transactions of the ASME*, 1955, **77**(6), 853–861.

[13] **Angeles, J., Hommel, G.,** and **Kovács, P.** *Computational Kinematics*, 1993 (Kluwer Academic Publishers B. V., Dordrecht, The Netherlands).

[14] **Dahlquist, G.** and **Björck, Å.** *Numerical methods*, 1974 (Prentice-Hall, Englewood Cliffs, NJ, USA).

[15] **Yang, A.** and **Freudenstein, F.** Application of dual-number quaternion algebra to the analysis of spatial mechanisms. *Journal of Applied Mechanics*, 1964, **31**(2), 300–308.

[16] **Angeles, J.** and **Rojas, A.** An optimization approach to the branching problem of plane linkage synthesis. In Proceedings of the 6th World Congress on the Theory of Machines and Mechanisms, 1983 December 15-20, New Delhi, India, Wiley Eastern Limited 1983, Vol. 1, 120–123.

[17] **Gupta, K. C.** and **Beloiu, A. S.** Branch and circuit defect elimination in spherical four-bar linkages. *Mechanism and Machine Theory*, 1998, **33**(5), 491–504.

[18] **Chiang, C. H.** *Kinematics of Spherical Mechanisms*, 1988 (Cambridge University Press, Cambridge, UK).

APPENDIX A

Notation

$4_{\mathrm{A}}, 5_{\mathrm{A}}$	points within the R_{A} region in Example 2
a	non-zero length representing the scale of the isotropic 3R robot
A	first revolute joint of a spherical four-bar linkage
$\mathbf{A}$	$(N-1) \times (N-3)$ matrix
$\bar{\mathbf{A}}$	$(N-1) \times (N-2)$ matrix
$\boldsymbol{b}$	$(N-1)$-dimensional vector
B	second revolute joint of a spherical four-bar linkage
cx_i	cosine of x_i
C	third revolute joint of a spherical four-bar linkage
d	output sliding of the spatial RCCC linkage
d_0	reference output sliding
D	fourth revolute joint of a spherical four-bar linkage
e_R	residual error of the nonlinear least-square minimization algorithm
E_{i+}, E_{i-}	percentage error amplification of a perturbed solution
g_i, g_{ji}	generic nonlinear functions
h_i	constant coefficient of the inverse positioning problem of serial decoupled 6R robots
k_i	ith Freudenstein parameter of a four-bar linkage
M	number of bivariate equations
N	number of precision points

O	intersection point of the axes of rotation of a spherical four-bar linkage
p_i	independent design parameter of the linkage
$\boldsymbol{p}$	$(N-3)$-dimensional vector
$\bar{\boldsymbol{p}}$	$(N-2)$-dimensional vector
P_{3i}, P_{4i}	points in the contour plot of Example 1 where three and four contours intersect
$P(x)$	20th-degree polynomial in x
$\tilde{P}(x)$	perturbed version of $P(x)$
R_i	original root solution
R_{i+}, R_{i-}	perturbed root solutions
$\mathrm{R_A}$	region in the contour plot of Example 2 where all the contours almost overlap
sx_i	sine of x_i
t	parameter in the parametric form of the hyperbolic and the Archimedean spirals
α_i	ith-link dimension of a spherical four-bar linkage
θ_i	joint angle of serial decoupled 6R robots
τ_3	tangent of the third joint half-angle
ψ	input angle of the linkage
ψ_0	reference input angle
ϕ	output angle of the linkage
ϕ_0	reference output angle
$\mathcal{C}_i$	original contour
$\mathcal{C}_{i+}, \mathcal{C}_{i-}$	perturbed contours

APPENDIX B

The implementation of the *refinement algorithm* outlined in Subsection 3.1 using the Newton-Gauss method [14] is described here. The method is used to find iteratively the least-square approximation of a nonlinear overdetermined system of equations; at each iteration, the least-square approximation of a linear overdetermined system is computed.

Problem statement: Find an approximate solution to system of nonlinear equations

$$\boldsymbol{f}(\boldsymbol{x}) = \mathbf{0}, \tag{42}$$

where $\boldsymbol{x}$ is a n-dimensional vector and $\boldsymbol{f}$ is a m-dimensional vector, with $m > n$, that verifies those equations with the least-square error, i.e.,

$$\boldsymbol{f}^T\boldsymbol{f} \quad \to \quad \min_{\boldsymbol{x}} . \tag{43}$$

In general, no vector $\boldsymbol{x}$ can be found that verifies all the m scalar equations (42). However, approximations can be found that minimize the least-square error of the approximation.

Solution: first, an initial guess $\boldsymbol{x}^0$ of $\boldsymbol{x}$ has to be given. In our case, this guess is given by the intersections roughly estimated by inspection on the contour plots. Then, a sequence $\boldsymbol{x}^1, \boldsymbol{x}^2, \boldsymbol{x}^3, \ldots,$ is obtained, such that

$$\boldsymbol{x}^{i+1} = \boldsymbol{x}^i + \Delta\boldsymbol{x}^i, \tag{44}$$

with $\Delta\boldsymbol{x}^i$ found as the least-square approximation of an overdetermined linear system, namely,

$$\mathbf{J}(\boldsymbol{x}^i)\Delta\boldsymbol{x}^i = -\boldsymbol{f}(\boldsymbol{x}^i) \tag{45}$$

with $\mathbf{J}$ defined as the $m \times n$ Jacobian matrix of the vector function $\boldsymbol{f}(\boldsymbol{x})$, i.e.,

$$\mathbf{J}(\boldsymbol{x}) = \frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \boldsymbol{x}}. \tag{46}$$

Therefore, in *symbolic* form,

$$\Delta \boldsymbol{x}^{i} = -\left(\mathbf{J}^{T}\mathbf{J}\right)^{-1}\mathbf{J}^{T}\boldsymbol{f} \tag{47}$$

The procedure stops when a convergence criterion,

$$\|\Delta \boldsymbol{x}\| < \epsilon, \tag{48}$$

is met, with ϵ denoting a prescribed tolerance.

Therefore, the algorithm returns the *local optimum value* $\boldsymbol{x}^{*}$ closest to the initial guess $\boldsymbol{x}^{0}$ and truncated to the desired accuracy ϵ that best approximates the overdetermined system given by Eq. (42) in the least-square sense.

Remarks: It can be shown that the second-order approximation of eq.(42) is negative-definite, and hence, the sequence of error-values *decreases monotonically.* That is, in the neighborhood of a stationary point $\Delta \boldsymbol{x} = \mathbf{0}$, the first-order approximation of $\boldsymbol{f}(\boldsymbol{x})$ is good enough, and hence, if the procedure converges, it does so to a minimum. However, convergence is not guaranteed. The algorithm may converge slowly or not at all if the initial guess is far from the minimum or the matrix $\mathbf{J}^{T}\mathbf{J}$ is ill-conditioned. In our case, the initial guess is always close-enough to a local minimum.

Prescribed Function	$\psi = 2 - (5/t)\sin t$ $\phi = 2 + (5/t)\cos t$	$\phi = \sqrt{2\psi}$	$\psi = (0.1)\, t\, \cos t + 0.15797399$ $\phi = (0.1)\, t\, \sin t + 0.02045874$
PRECISION POINTS (ψ_i, ϕ_i)			
1	(0.37890171, 0.55157387)	(0.20000000, 0.63245553)	(-0.16978585, -0.10231539)
2	(2.23900863, 0.50381853)	(0.40000000, 0.89442719)	(-0.04793556, -0.34560309)
3	(3.06530923, 1.53395445)	(0.60000000, 1.09544512)	(0.24936505, -0.46094304)
4	(2.78515796, 2.52299466)	(0.80000000, 1.26491106)	(0.59229088, -0.33305058)
5	(1.98665563, 2.79353860)	(1.00000000, 1.41421356)	(0.78788493, 0.03105150)
6	(1.41750916, 2.36032707)	(1.20000000, 1.54919334)	(0.68570557, 0.48034936)

	Our Method		**Our Method**		**Our Method**		**Method in [6]**	
COMPUTED DESIGN PARAMETERS								
α_1	2.7562506	1.3235236	1.5679253	1.3761275	1.5826845	0.6913904	1.5832970	0.6913904
α_2	2.6036225	0.9672887	0.4371065	0.6117110	2.6635344	0.3128503	2.8097423	0.3128503
α_3	0.3533623	0.9485493	1.9048276	2.2216431	1.8801156	0.4142372	1.8841588	0.4142372
α_4	0.8977034	1.7283503	2.3637434	2.0674146	1.7471203	0.6604776	1.7471203	0.6604776
ψ_0	1.4164239	-1.2111495	1.3632058	1.5000000	-1.5707963	0.4191038	-1.5707963	0.4191038
ϕ_0	-1.3210544	-0.4418615	1.2142519	1.2930765	1.5707963	1.0435117	1.5707963	1.0435117

Table 1: Results obtained for valid designs

Prescribed Function: $\psi = 2 - (5/t)\sin t, \quad \phi = 2 + (5/t)\cos t$							
No.	ψ_0	ϕ_0	e_R	k_1	k_2	k_3	k_4
1	**1.4164**	$\mathbf{-1.3211}$	$\mathbf{1.09 \times 10^{-15}}$	**2.4352**	**2.1199**	$\mathbf{-1.9277}$	**0.6234**
2	$\mathbf{-1.2111}$	$\mathbf{-0.4419}$	$\mathbf{3.95 \times 10^{-15}}$	$\mathbf{-0.7488}$	**0.7084**	**0.2493**	$\mathbf{-0.1569}$
3	-1.2172	1.0146	2.03×10^{-15}	-6.2289	10.4235	-7.1700	-11.4489
4	0.2733	-0.8098	1.18×10^{-15}	-1.1498	-1.1059	1.3641	-1.1255
5	0.7198	-1.5222	0.60×10^{-15}	18.0309	21.3054	-30.8173	37.2972

Table 2: Numerical values of the intersection points for Example 1

Prescribed Function: $\phi = \sqrt{2\psi}$							
No.	ψ_0	ϕ_0	e_R	k_1	k_2	k_3	k_4
1	0.6452	0.9555	1.40×10^{-16}	-1.3623	0.9252	-0.9251	-1.3624
2	1.4027	-1.1015	1.53×10^{-16}	1.2343	-0.1314	-0.8487	-1.0393
3	**1.3632**	**1.2143**	$\mathbf{1.42 \times 10^{-16}}$	$\mathbf{-0.9583}$	$\mathbf{-0.2436}$	**0.0020**	$\mathbf{-0.7124}$
$\mathbf{4}_{\mathrm{A}}$	**1.5000**	**1.2931**	$\mathbf{3.68 \times 10^{-8}}$	$\mathbf{-0.9773}$	$\mathbf{-0.6696}$	**0.1733**	$\mathbf{-0.4765}$
5_{A}	-0.9558	-1.3696	2.64×10^{-9}	-0.5251	-1.0336	-0.1174	-1.6763

Table 3: Numerical values of the intersection points for Example 2

Prescribed Function: $\psi = (0.1)\, t\, \cos t + 0.15797399, \quad \phi = (0.1)\, t\, \sin t + 0.02045874$

No.	ψ_0	ϕ_0	e_R	k_1	k_2	k_3	k_4
1	$\mathbf{-1.5708}$	$\mathbf{1.5708}$	$\mathbf{2.32 \times 10^{-16}}$	$\mathbf{0.9315}$	$\mathbf{-0.3146}$	$\mathbf{-0.0117}$	$\mathbf{-0.1754}$
2	$\mathbf{0.4191}$	$\mathbf{1.0435}$	$\mathbf{2.21 \times 10^{-16}}$	$\mathbf{-1.5375}$	$\mathbf{1.3953}$	$\mathbf{0.7412}$	$\mathbf{0.7897}$
3	-0.5032	0.1965	3.75×10^{-16}	-129.716	131.338	130.543	132.166
4	-0.2541	0.2954	3.71×10^{-16}	-116.581	126.933	119.165	129.660
5	-0.0997	0.3727	3.68×10^{-16}	-21.3945	24.2833	21.1979	24.0815
6	-0.3085	-0.0785	2.27×10^{-16}	-5.5586	-1.8540	-3.6938	-12.3156
7	-0.6784	0.1307	3.90×10^{-16}	-26.9659	27.0212	27.0733	27.1457

Table 4: Numerical values of the intersection points for Example 3

LIST OF FIGURE CAPTIONS

Figure 12: Plot associated with the generation of $\phi = \sqrt{2\psi}$: second solution

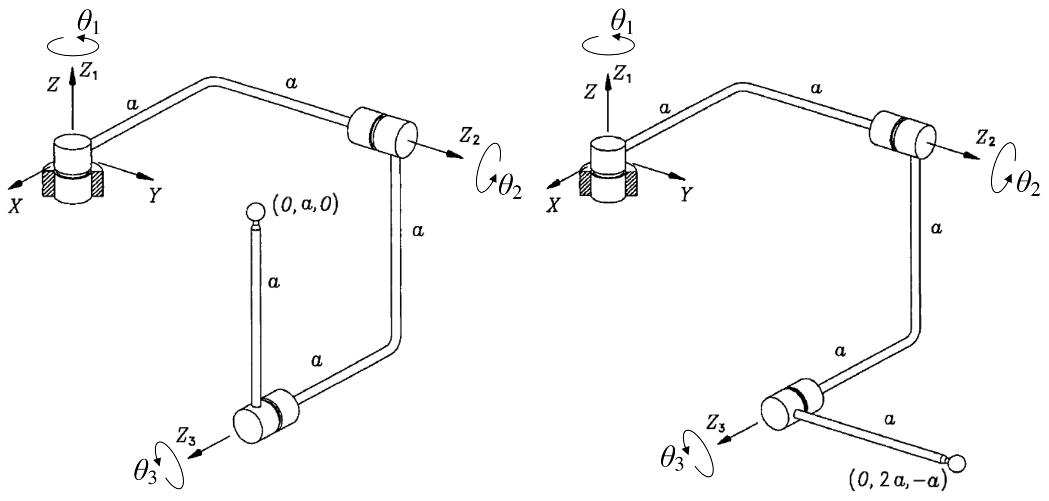
Figure 13: Plot of the five contours for Example 3

Figure 14: Plot of the five contours in the range $[(-\pi/2, -\pi/2), (\pi/2, \pi/2)]$ for the Example 3

Figure 15: Plot associated with the generation of $\{\psi = (0.1)\, t\ \cos t{+}0.15797399,\ \phi = (0.1)\, t\ \sin t\ \ + 0.02045874\}$: first solution

Figure 16: Plot associated with the generation of $\{\psi = (0.1)\, t\ \cos t{+}0.15797399,\ \phi = (0.1)\, t\ \sin t\ \ + 0.02045874\}$: second solution

FIGURES (1-16)



(a) (b)

Figure 1: An orthogonal 3R robot: configurations $(0, a, 0)$ and $(0, 2a, -a)$

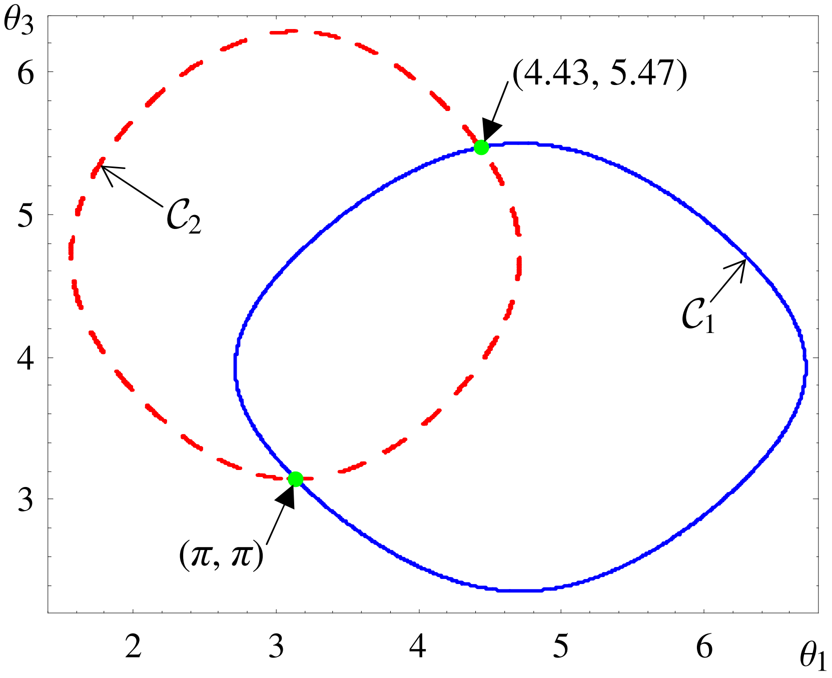
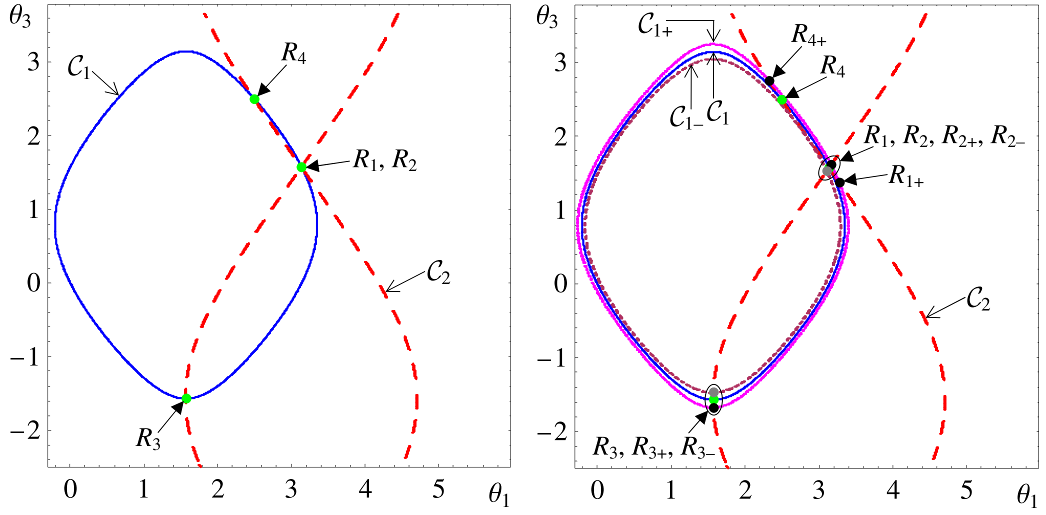


Figure 2: Contours producing the two real solutions of the deflation example



(a) (b)

Figure 3: An example of ill-conditioning

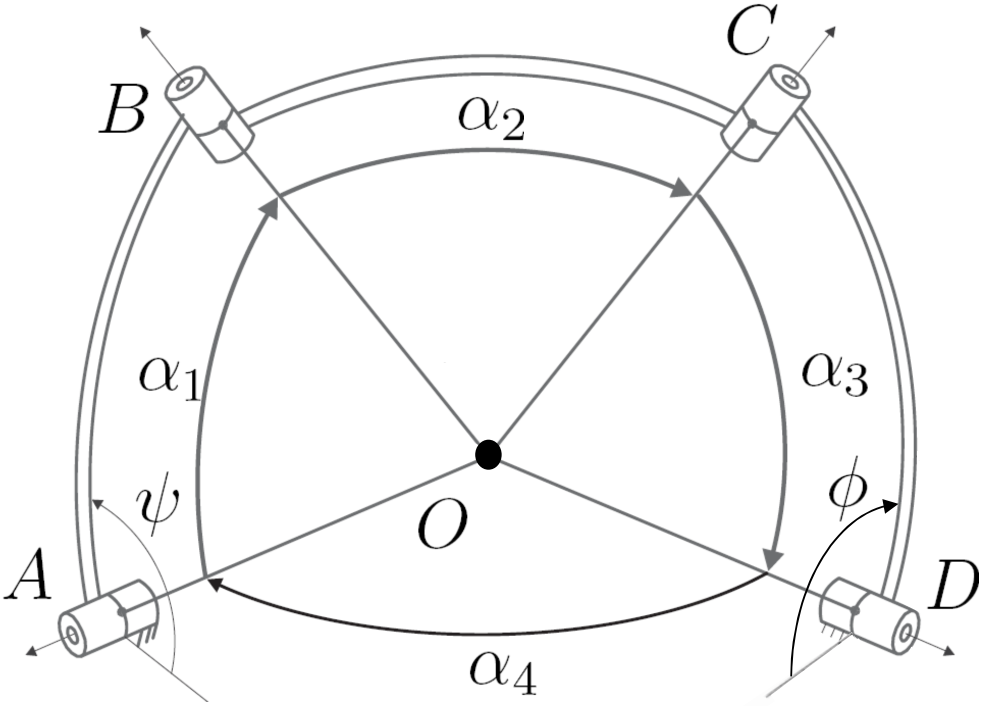


Figure 4: A general spherical four-bar linkage

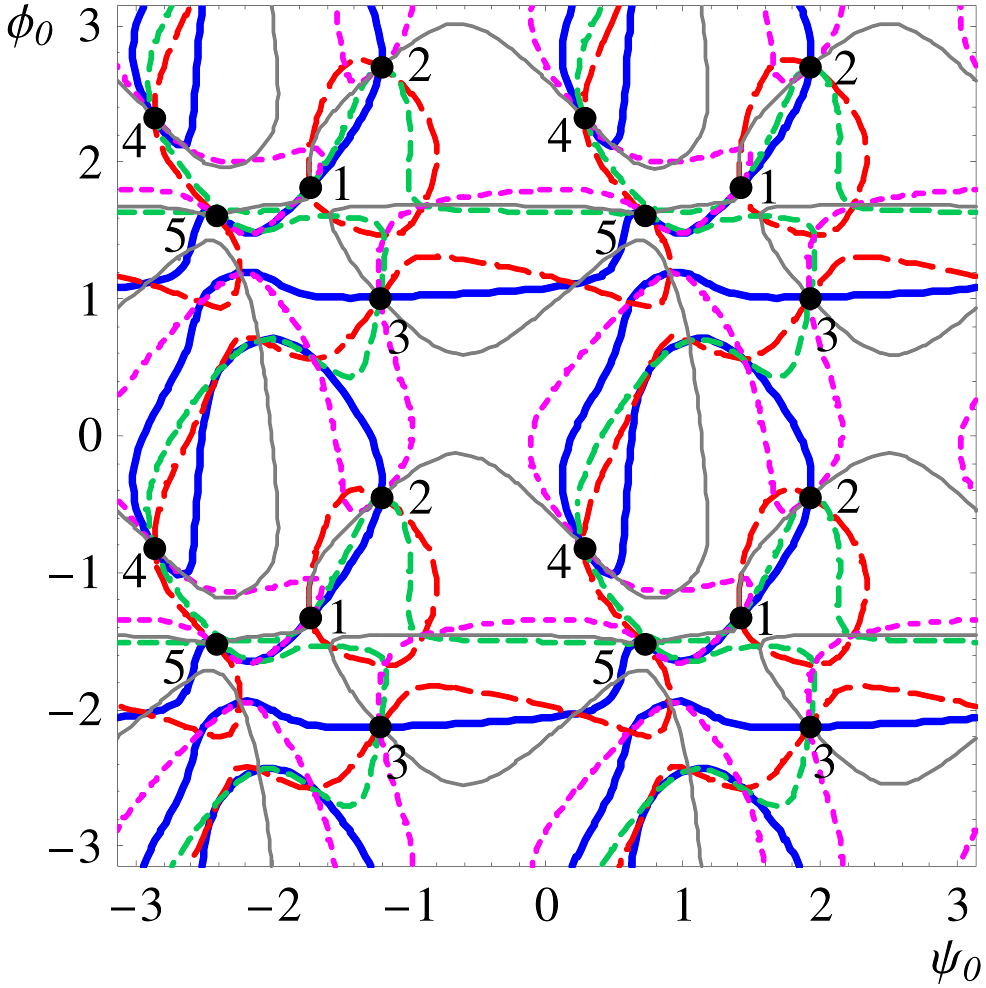


Figure 5: Plot of the five contours for Example 1

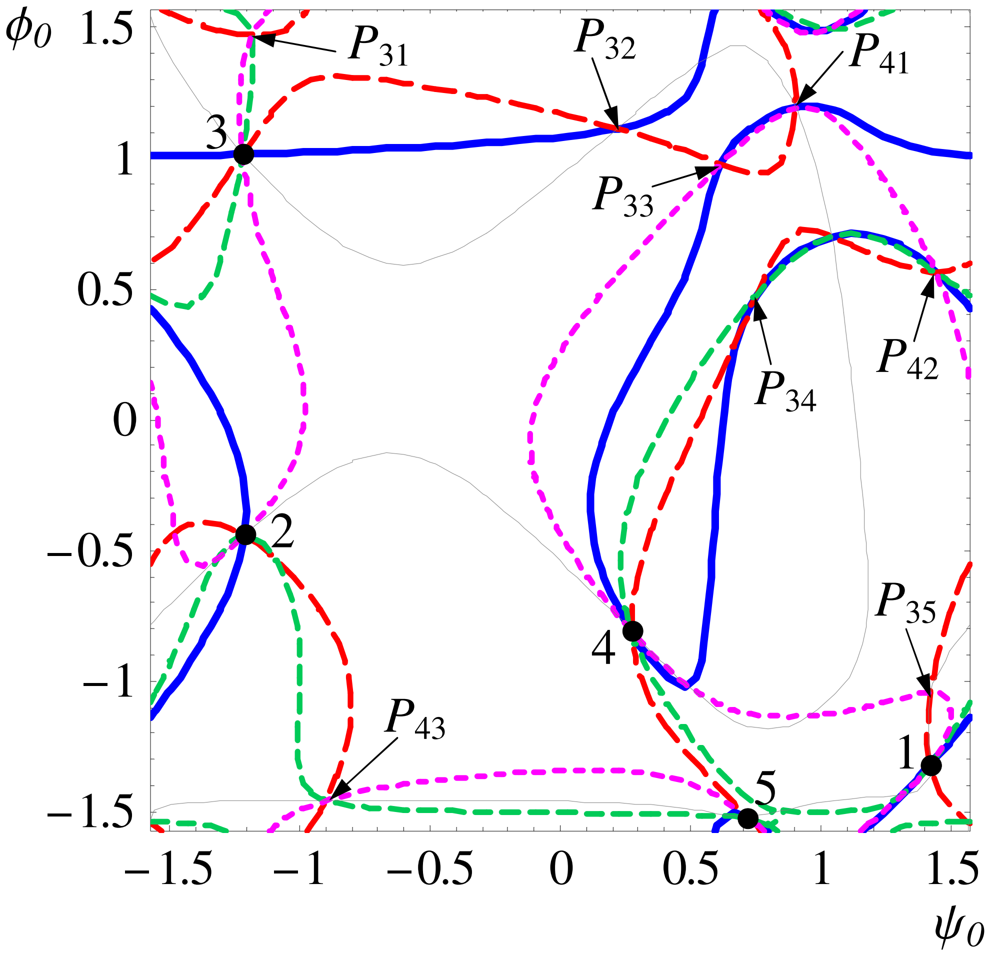


Figure 6: Plot of the five contours in the range $[(-\pi/2, -\pi/2), (\pi/2, \pi/2)]$ for Example 1

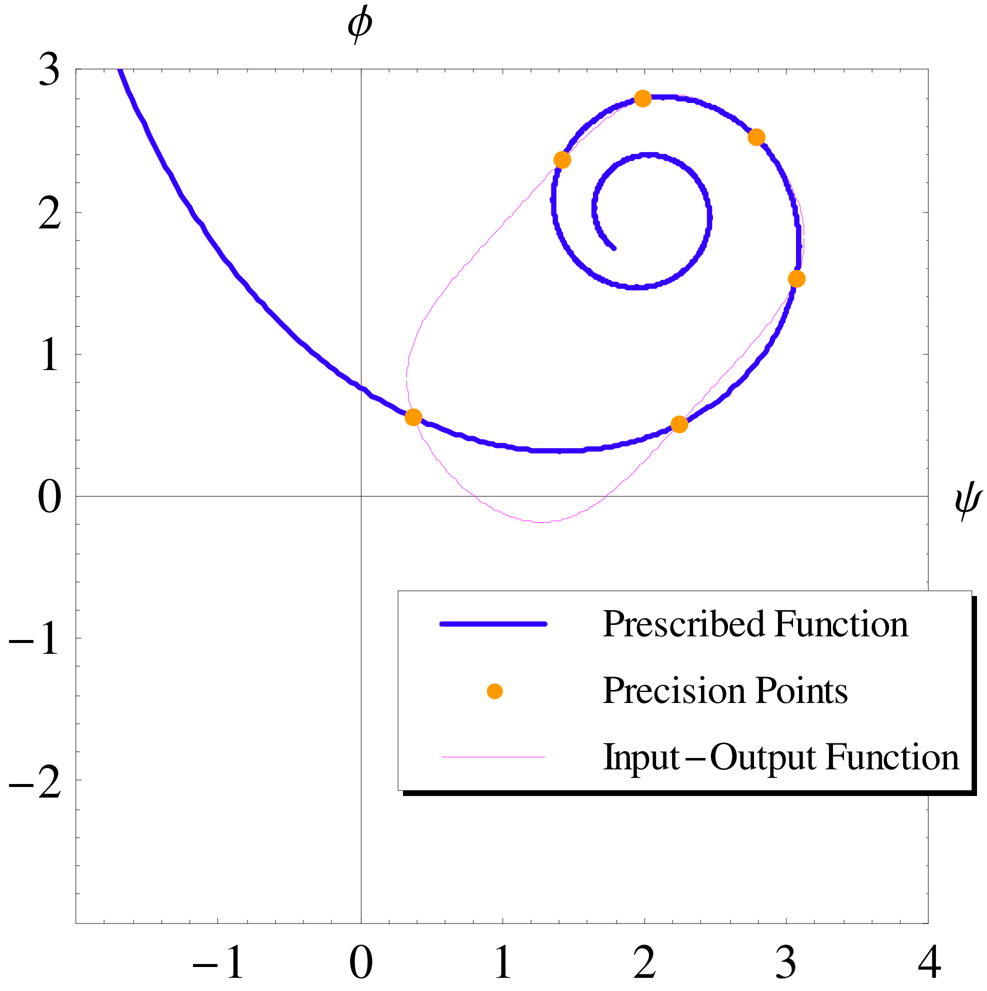


Figure 7: Plot associated with the generation of $\{\psi = 2 - (5/t)\,\sin t,\ \phi = 2 + (5/t)\,\cos t\}$: first solution

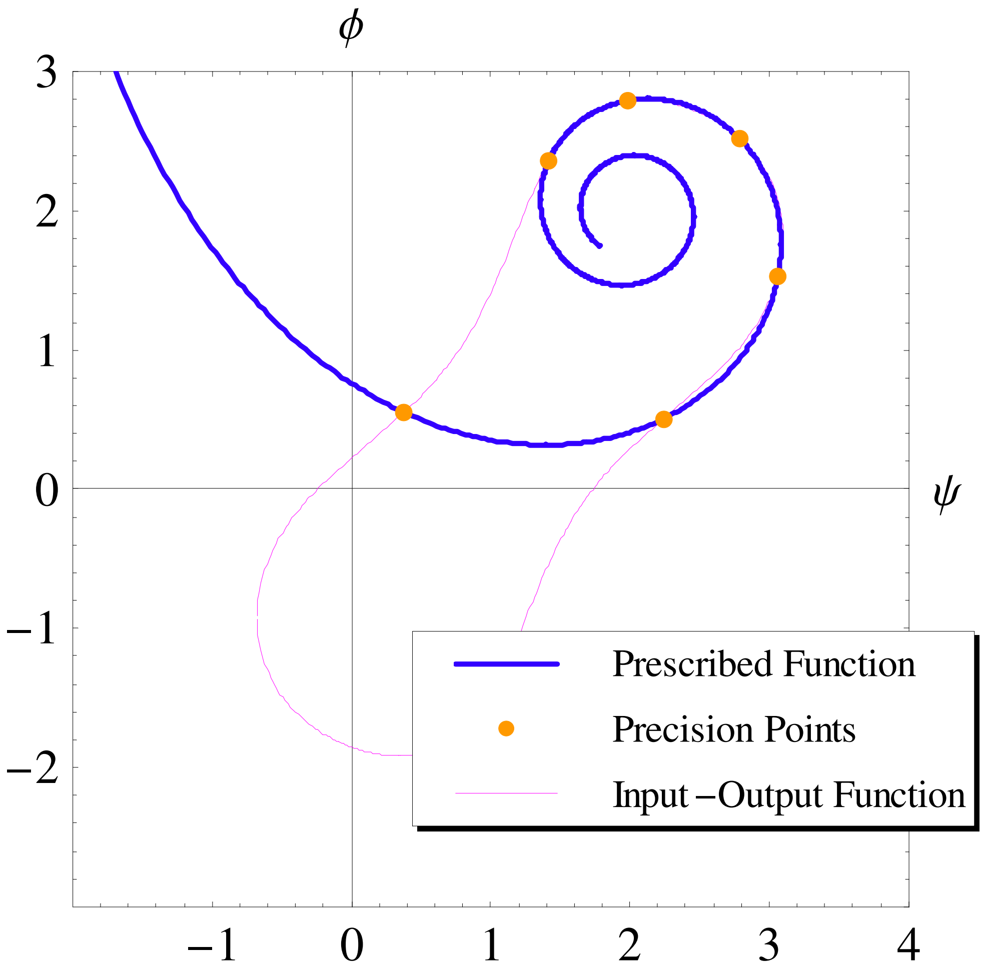


Figure 8: Plot associated with the generation of $\{\psi = 2 - (5/t)\ \sin t,\ \phi = 2 + (5/t)\ \cos t\}$: second solution

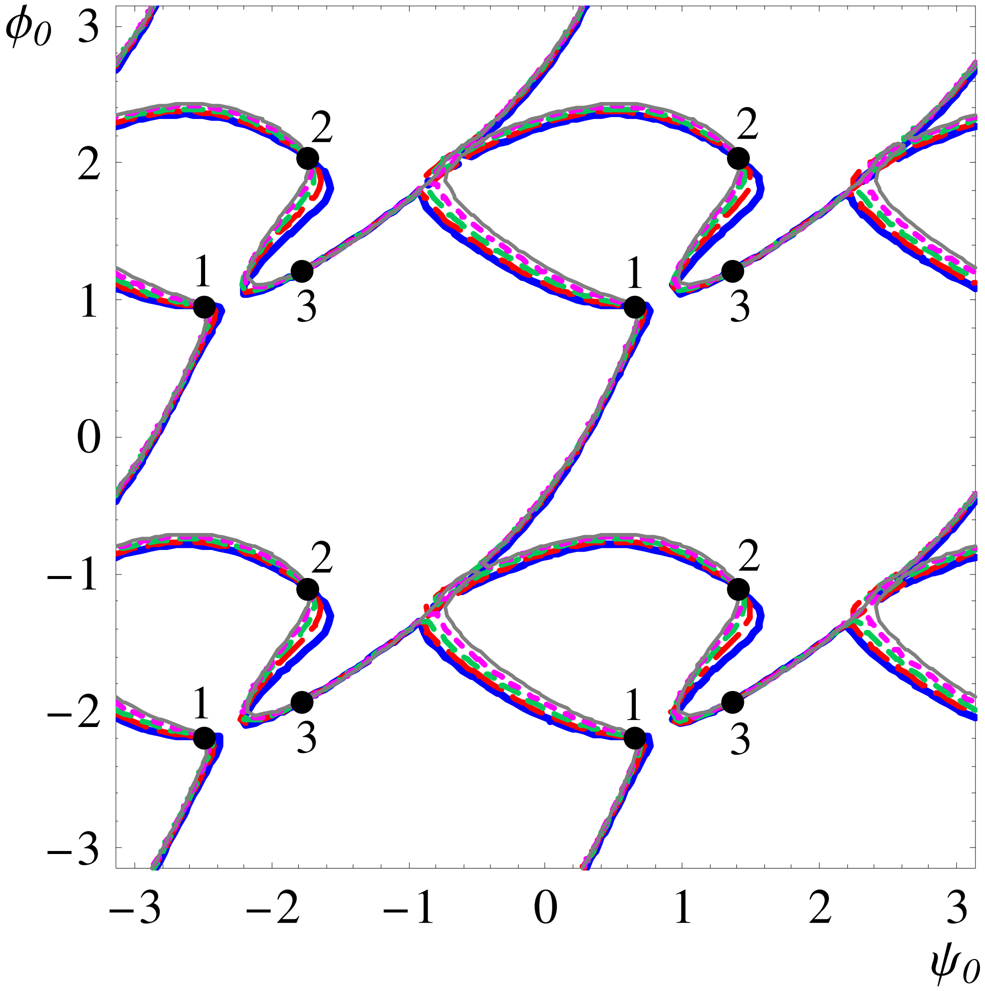


Figure 9: Plot of the five contours for Example 2

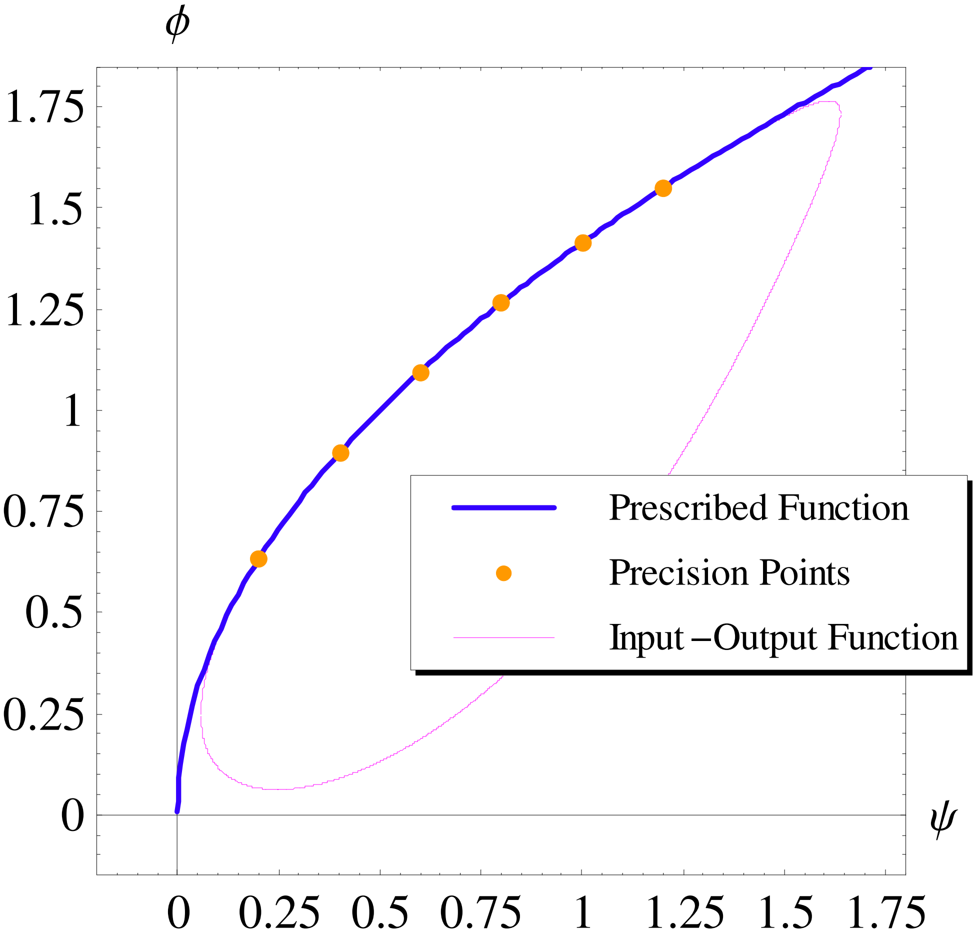


Figure 10: Plot associated with the generation of $\phi = \sqrt{2\psi}$: first solution

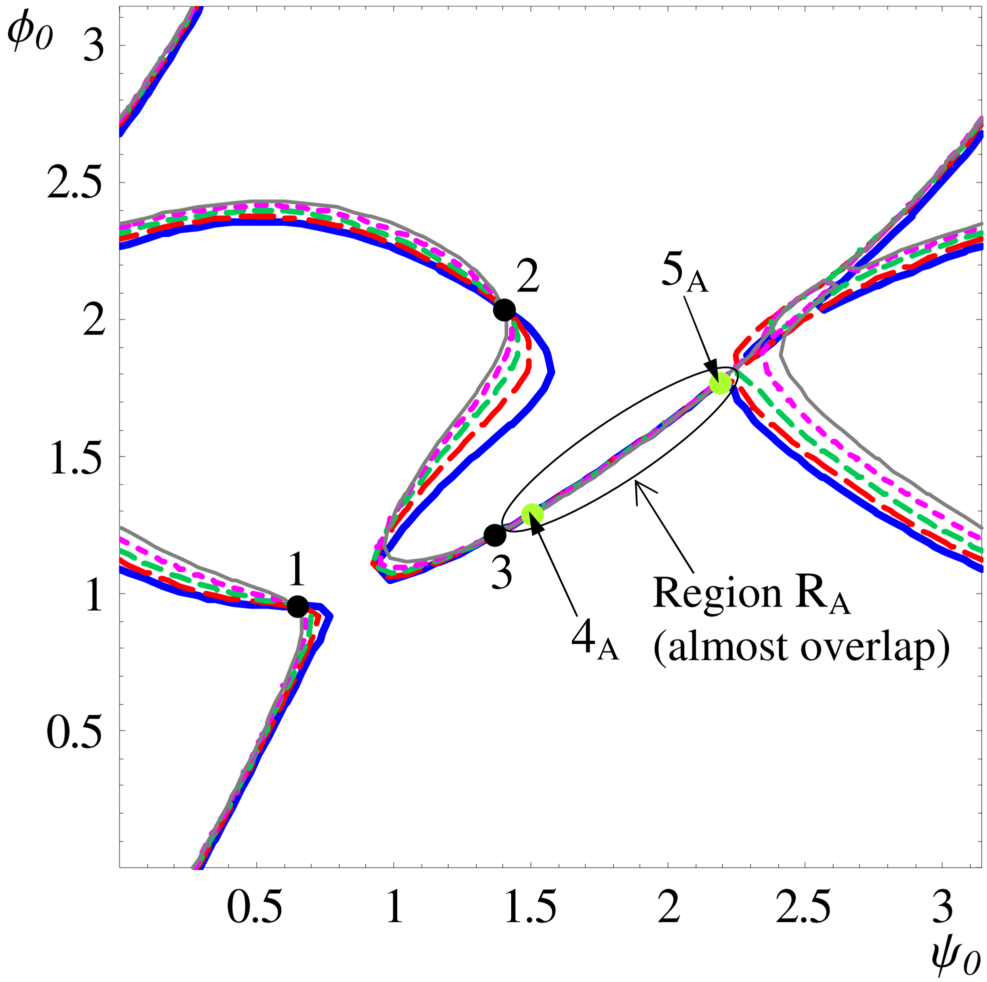


Figure 11: Plot of the five contours in the range $[(0,0),(\pi,\pi)]$ for Example 2

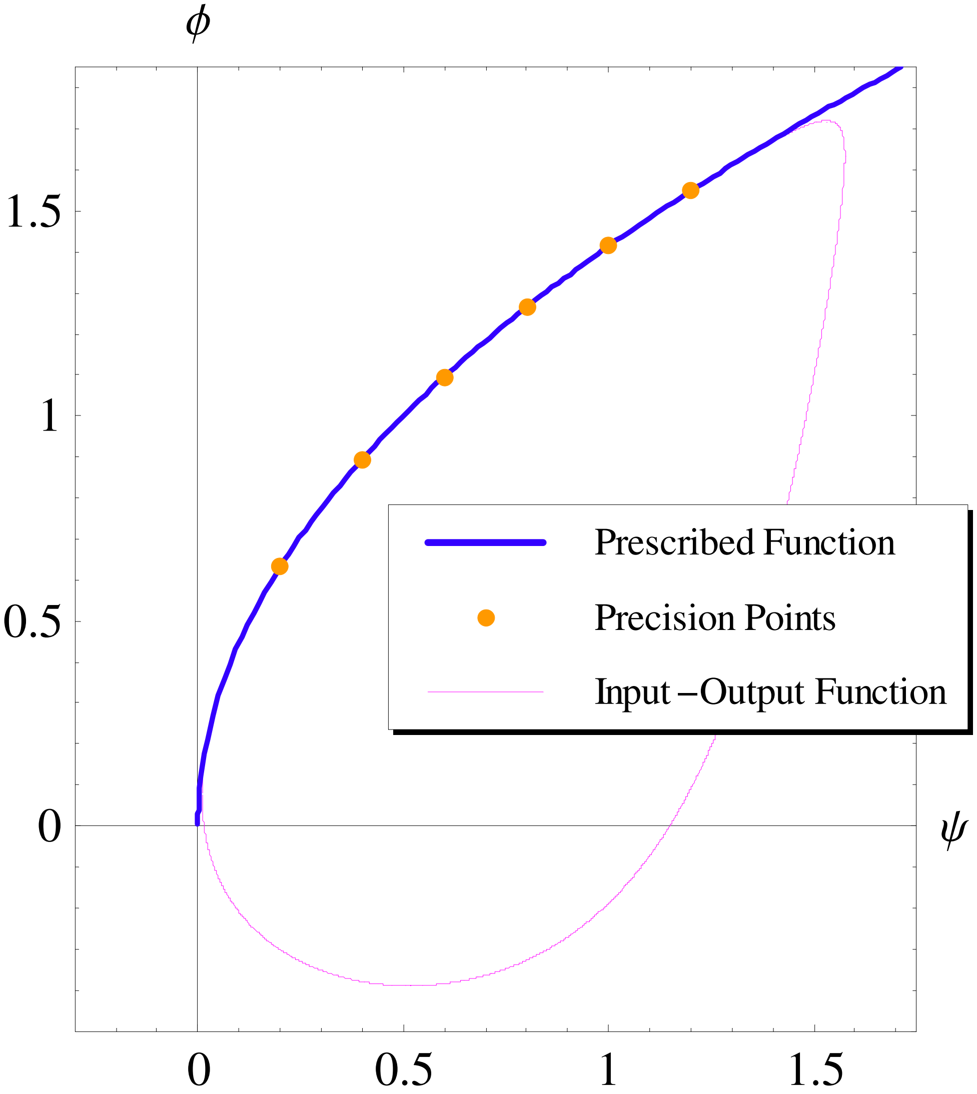


Figure 12: Plot associated with the generation of $\phi = \sqrt{2\psi}$: second solution

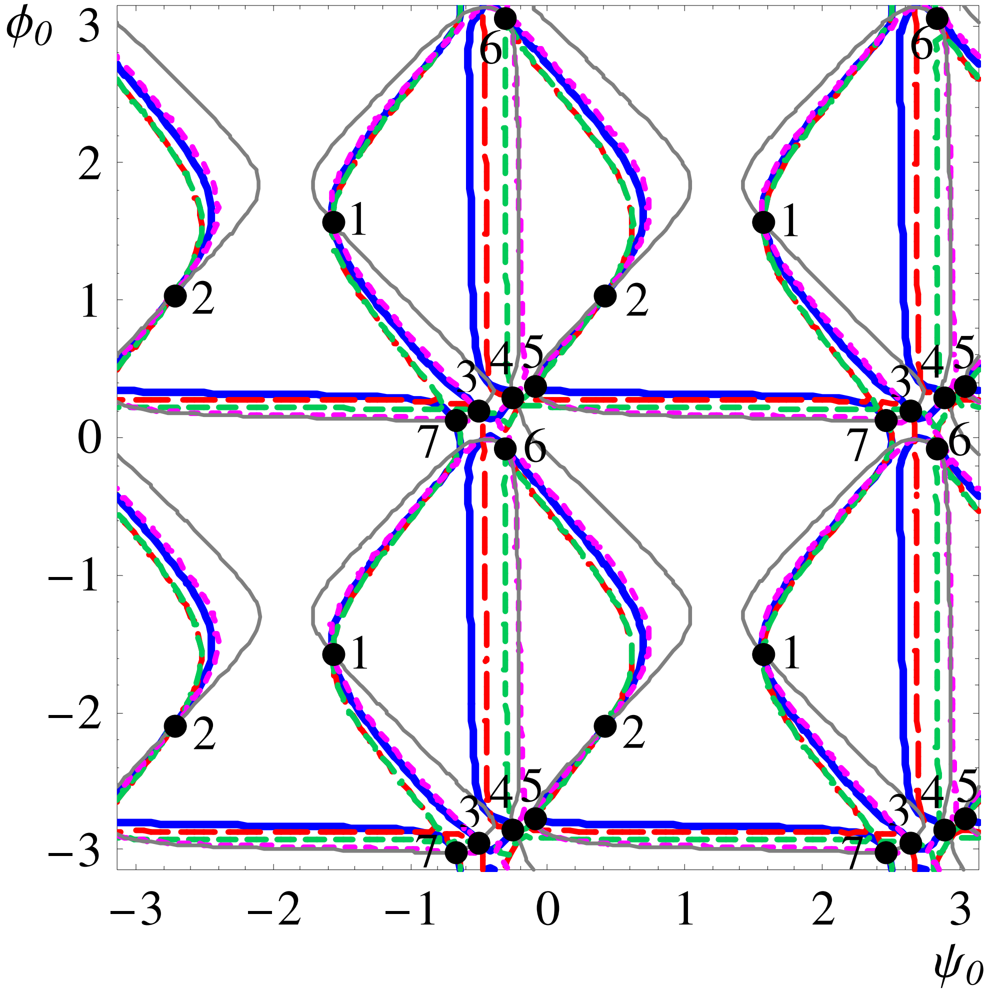


Figure 13: Plot of the five contours for Example 3

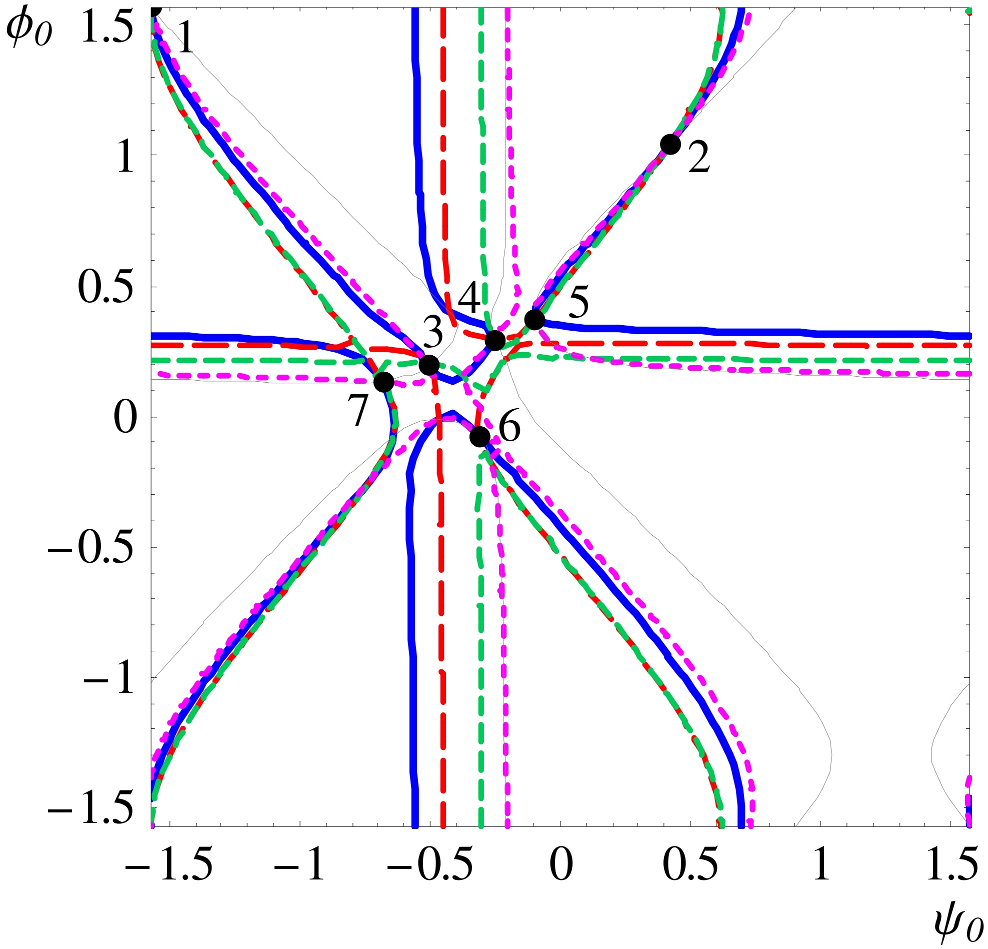


Figure 14: Plot of the five contours in the range $[(-\pi/2, -\pi/2), (\pi/2, \pi/2)]$ for Example 3

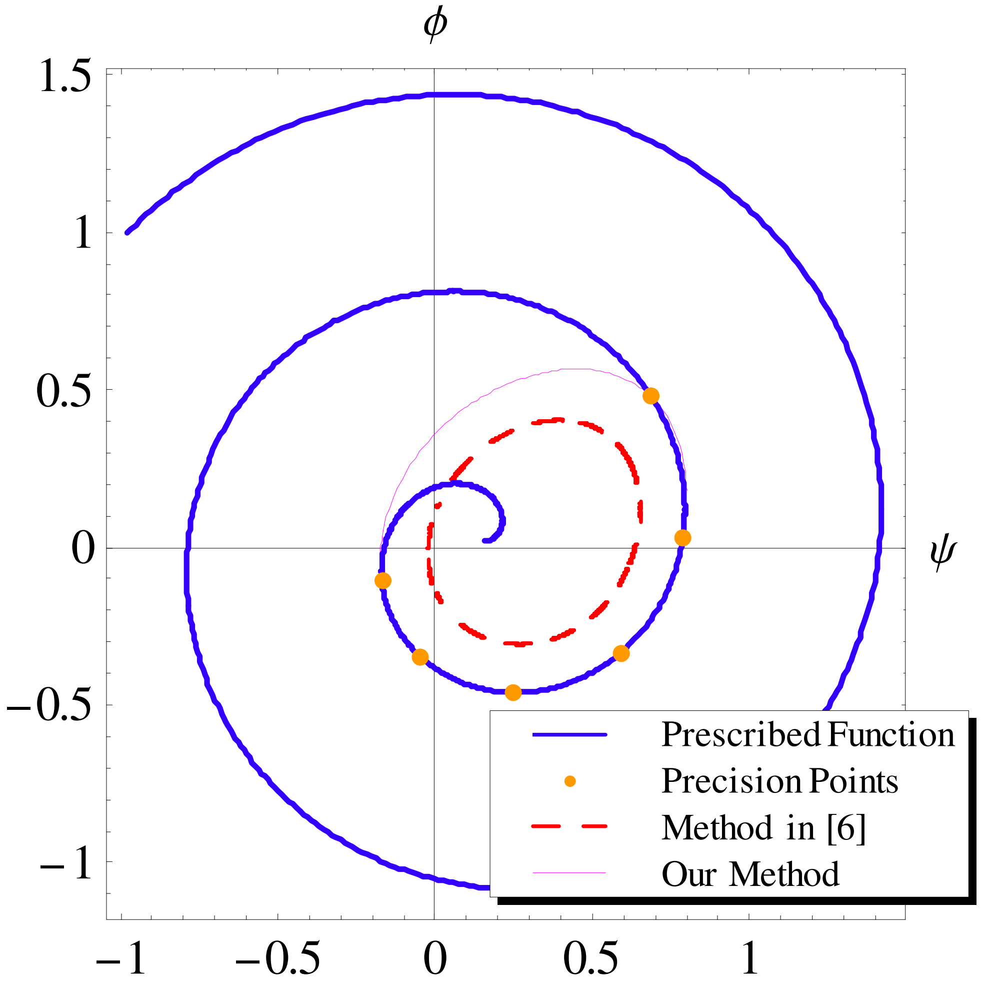


Figure 15: Plot associated with the generation of $\{\psi = (0.1)\,t\,\cos t + 0.15797399,\ \phi = (0.1)\,t\,\sin t\ + 0.02045874\}$: first solution

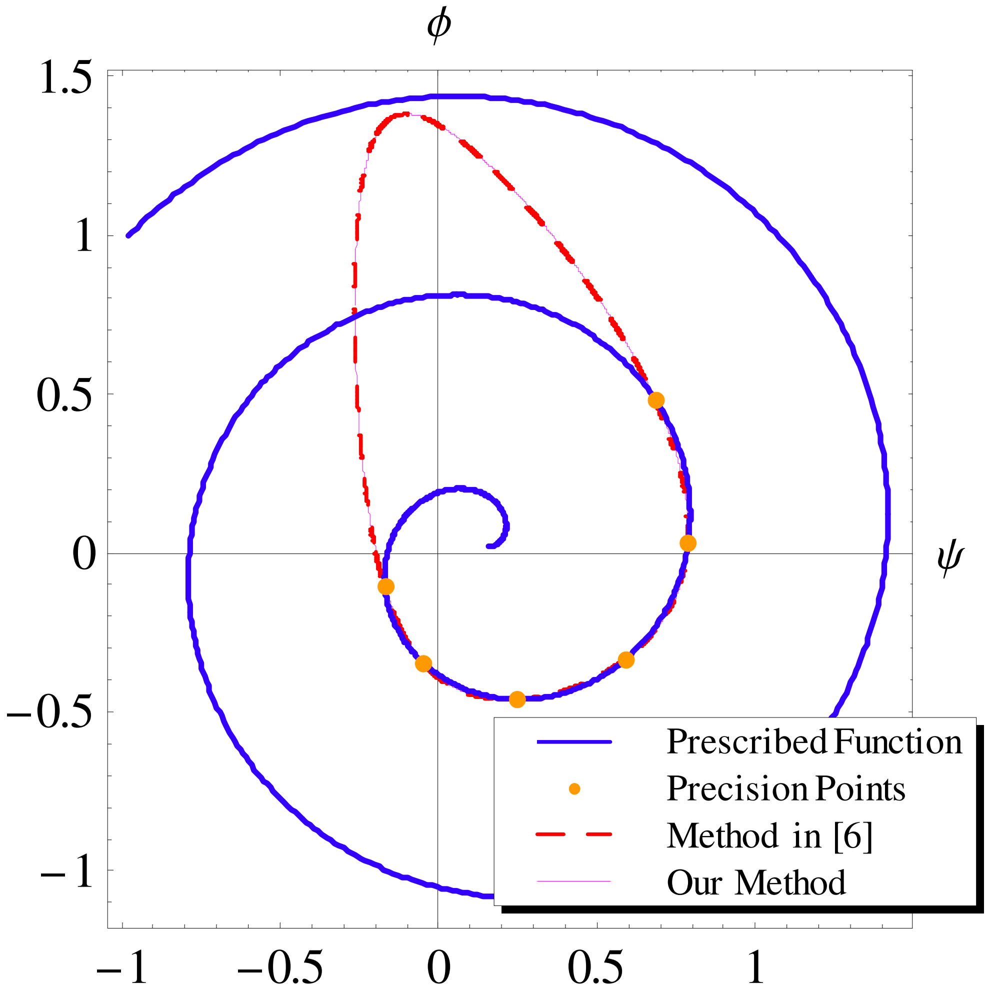


Figure 16: Plot associated with the generation of $\{\psi = (0.1)\, t \cos t + 0.15797399,\ \phi = (0.1)\, t \sin t\ + 0.02045874\}$: second solution