

Chapter 6

Paraxial ray tracing in the position/angle domain

A ray field in the position/angle domain has only a limited amount of geometrical spreading and its corresponding ray field map is single-valued, regardless of the complexity of the medium. These are exactly the type of circumstances that – in the spatial domain – allow the use of paraxial ray methods for the calculation of ray field maps. This observation suggests the use of paraxial ray methods in the position/angle domain as well.

A paraxial ray method is proposed that can be used for the calculation of ray field maps in the position/angle domain in smooth media of arbitrary complexity. The algorithm is both efficient and simple. The mapping from ray field coordinates to position/angle coordinates is performed by averaging the contributions of rays passing through a small neighbourhood around each of the map's grid points. This is different from the usual approach in paraxial ray tracing in which only the nearest passing ray is used. The advantage of averaging is that the local errors are more smoothly distributed which leads to more stable estimation of derivatives in further applications of the ray field map.

6.1 Introduction

In Chapter 5 the applicability of the proposed ray field construction method in the position/angle domain was tested. The unfortunate conclusion was that this method is not suitable for that domain, due to the type of deformation in the ray field structure. In the position/angle domain this deformation is dominantly shear-like while the ray field construction algorithm is primarily designed for expanding ray fields in the spatial domain.

By looking at the ray field map jacobian (3.28) three important observations can be made. First, the jacobian does not vanish anywhere, so the mapping from

ray field coordinates to position/angle coordinates is regular and single-valued.

Second, the geometrical spreading in the position/angle domain is limited. A look at (3.30) reveals that in 3-D media the geometrical spreading in the position/angle domain is roughly proportional to the third power of the local phase velocity. Given that the ray field map in the position/angle domain is 5-D, the spreading per dimension is limited. Moreover, the local ray density is approximately known a priori, from the values of the local phase velocity.

Third, each ray field emitted from a single point source can exhibit large geometrical spreading in the spatial domain. The combination of spatial divergence of neighbouring rays with the absence of geometrical spreading in the position/angle domain can only be explained if the deformation of the ray field structure in the position/angle domain is shear-like, as was indeed observed in Section 5.3.3.

Two of the most important incentives for the development of the wave front construction methods for ray tracing in complex media were the presence of multi-pathing and the large variations in geometrical spreading. In simpler media, in absence of these complicating features, paraxial ray methods (Červený et al., 1984; Beydoun and Keho, 1987) are generally more efficient, because they do not require the maintenance of a geometrical structure for the ray field. Since in the position/angle domain both multi-pathing and extensive geometrical spreading are absent and the ray field construction method fails due to the shear-like deformations, it is natural to investigate the applicability of paraxial ray methods.

In this chapter a paraxial ray tracing and mapping algorithm is developed in the position/angle domain. In paraxial ray tracing the geometrical structure of the ray field, i.e., the network of connections between neighbouring rays, is not available. Therefore it is not possible to calculate the ray field map by means of (inverse) interpolation as in ray field construction (see Chapter 5). In classical paraxial ray tracing in the spatial domain the mapping is usually performed by extrapolation from the nearest neighbouring ray. In Section 6.2 an alternative mapping method is proposed, which is based on an averaging integral. The paraxial ray method for the position/angle domain itself is introduced in Section 6.3.

6.2 Regularisation by averaging

The task of finding the ray field map on a regular grid can be seen as a problem of regularisation. The points of evaluation of the ray field form an irregular distribution of data points in the mapping domain. One way to perform the regularisation is by means of interpolation.

Unfortunately, interpolation is in general much more complicated in irregular, than in regular data distributions. One of the main difficulties is the determination of a geometric structure on the distribution, which is essential in order to determine the data points that contribute to a given interpolation point (see also Appendix D.2). In the ray field construction method of Chapter 5 the geometrical structure is maintained throughout the ray tracing process. For the paraxial ray method

developed in this chapter no such structure is available. It is then more convenient to perform regularisation by an averaging integral than by interpolation.

6.2.1 Averaging integrals

Continuous formulation

The integral approach to data regularisation is based on convolution of the function to be approximated with an averaging or smoothing kernel. In one dimension a regularisation \mathcal{R}_0 for a function $f(x)$ may be expressed as

$$\mathcal{R}_0[f(x)](x) = \int_{-\infty}^{\infty} K(\xi)f(x - \xi)d\xi. \quad (6.1)$$

If kernel $K(x)$ is defined to have a unit integral:

$$\int_{-\infty}^{\infty} K(\xi)d\xi = 1, \quad (6.2)$$

and a vanishing first order moment:

$$\int_{-\infty}^{\infty} K(\xi)\xi d\xi = 0, \quad (6.3)$$

then integral (6.1) may be regarded as an approximate representation integral that is accurate to first order. This can be verified by inserting a first order polynomial:

$$\mathcal{R}_0[c_0 + c_1x](x) = c_0 + c_1x. \quad (6.4)$$

Note that this agrees with the common notion that smoothing preserves linear features.

In multiple dimensions an equivalent regularisation may be defined:

$$\mathcal{R}_0[f(\mathbf{x})](\mathbf{x}) = \iint_{\text{supp}(K)} K(\boldsymbol{\xi})f(\mathbf{x} - \boldsymbol{\xi})d\boldsymbol{\xi}, \quad (6.5)$$

where \mathbf{x} and the double integral sign stand for arbitrary dimension. The infinite limits of the integral are replaced by $\text{supp}(K)$ in order to denote that in practice K will have a localised support.

The constraints on kernel K generalise analogously:

$$\iint_{\text{supp}(K)} K(\boldsymbol{\xi})d\boldsymbol{\xi} = 1, \quad \text{and} \quad (6.6)$$

$$\iint_{\text{supp}(K)} K(\boldsymbol{\xi})\boldsymbol{\xi}d\boldsymbol{\xi} = \mathbf{0}. \quad (6.7)$$

Regularisation based on the approximate representation integral (6.5) is performed by evaluating the integral numerically using the available data points. A particular advantage of this approach, as opposed to regularisation by interpolation, is that it does not require as much a priori knowledge about the geometrical arrangement of the data points. A disadvantage, however, is that the numerical evaluation of the integral introduces additional numerical errors.

Discrete formulation

A discrete analogue of (6.5), based on the M data points \mathbf{x}_i ($i = 1 \dots M$) within the support of $K(\mathbf{x} - \mathbf{x}_i)$, may be expressed as

$$\mathcal{R}_0^D[f(\mathbf{x})](\mathbf{x}) = \sum_{i=1}^M \alpha_i f(\mathbf{x}_i), \quad (6.8)$$

where the weights α_i are yet to be determined.

It is obvious that an arbitrary distribution of data points does not offer a high degree of accuracy control in the numerical evaluation of an integral. It should be noted, however, that in order for (6.5) to serve as an approximate representation integral, the actual shape of kernel K is less critical than the validity of accuracy constraints (6.6) and (6.7). These constraints have discrete analogues

$$\sum_{i=1}^M \alpha_i = 1, \quad \text{and} \quad (6.9)$$

$$\sum_{i=1}^M \alpha_i \mathbf{x}_i = \mathbf{x}, \quad (6.10)$$

respectively.

At this point it is interesting to note the similarities between this integral approach to regularisation, and interpolation in terms of barycentric coordinates, as described in Appendix D.2. In fact, barycentric interpolation can be seen as a special case of the method developed here, using only a minimal set of data points. In that case, the weights α_i , the barycentric coordinates, are uniquely determined by the constraints (D.8) and (D.9), which makes the use of an averaging kernel redundant.

Before discussing how to determine weights α_i , it is important to say a word about the practical implementation of this type of regularisation. The numerical evaluation of integral (6.5), through its discrete counterpart (6.8), is most conveniently implemented as a generalised (weighted) binning, by looping over the data points. Because of the local support of kernel K , it is easy to determine the range of grid points that each data point contributes to. Therefore, it is desirable to be able to determine the weight α_i for each contribution before knowing which or how many data points will contribute to the integral in the end. In practice this is

not entirely possible, but it is important that the amount of bookkeeping required for each grid point be kept minimal.

Determination of weights α_i

For a grid point denoted by \mathbf{x} , the relative contribution $\hat{\alpha}_i$ of data point \mathbf{x}_i is determined by kernel K :

$$\hat{\alpha}_i = K(\mathbf{x} - \mathbf{x}_i). \quad (6.11)$$

If an estimate of a local data point density $\rho(\mathbf{x}_i)$ is available it may be included in the relative weight:

$$\hat{\alpha}_i = \rho(\mathbf{x}_i)K(\mathbf{x} - \mathbf{x}_i). \quad (6.12)$$

This is the case, for example, if the averaging integral is calculated on a set of points that form a regular distribution in a domain other than \mathbf{x} , say \mathbf{y} . The data point density is then equal to the jacobian of transformation from \mathbf{y} to \mathbf{x} . This is used also in the paraxial ray method to be discussed in Section 6.3.

In the binning process the relative contributions are added, while keeping track of the sum of relative weights. Afterwards the discrete integral (6.8) is obtained using

$$\mathcal{R}_0^D[f(\mathbf{x})](\mathbf{x}) = \frac{\sum_{i=1}^M \hat{\alpha}_i f(\mathbf{x}_i)}{\sum_{i=1}^M \hat{\alpha}_i}, \quad (6.13)$$

which effectively sets the weights α_i for (6.8):

$$\alpha_i = \frac{\hat{\alpha}_i}{\sum_{i=1}^M \hat{\alpha}_i}. \quad (6.14)$$

The constraint (6.9) is automatically satisfied. The bookkeeping for each grid point is limited to a single number that accumulates the sum of relative weights.

The fulfillment of constraint (6.9) guarantees that the regularisation is exact only for constant functions. In order to be first order accurate, it is necessary to satisfy constraint (6.10) as well. Unfortunately this is much more difficult to achieve. Adapting the weights α_i in such a way that both constraints (6.9) and (6.10) are satisfied, requires extensive bookkeeping, which is to be avoided.

Mislocation

The degree of violation of constraint (6.10) can be seen as a measure of how well the continuous regularisation integral (6.5) is approximated by its discrete counterpart (6.8). The error may be interpreted as a *mislocation* of the regularisation estimate.

For a given set of data points \mathbf{x}_i and weights α_i that satisfy constraint (6.9), the first order moment $\delta\mathbf{x}$ with respect to grid point \mathbf{x} may be evaluated:

$$\delta\mathbf{x} = \sum_{i=1}^M \alpha_i (\mathbf{x}_i - \mathbf{x}). \quad (6.15)$$

If $\delta\mathbf{x}$ vanishes, constraint (6.10) is satisfied and the discrete regularisation integral (6.8) is accurate to first order. For α_i obtained using the procedure outlined above, this will, in general, not be the case. Instead, it is easy to see that the first order moment does vanish if it is calculated with respect to the point $\mathbf{x} + \delta\mathbf{x}$:

$$\sum_{i=1}^M \alpha_i (\mathbf{x}_i - \mathbf{x} - \delta\mathbf{x}) = \mathbf{0}. \quad (6.16)$$

In other words, the combination of data points \mathbf{x}_i and weights α_i provides a first order accurate estimator for the function value $f(\mathbf{x} + \delta\mathbf{x})$, rather than for $f(\mathbf{x})$.

6.2.2 Kernels and mislocation

Mislocation is an important source of error in the integral regularisation approach, as will be demonstrated in Section 6.2.3. Factors that influence the amount of mislocation are the size and the shape of averaging kernel K , the number of data points within the support of K , and the spatial distribution of the data points. A crucial piece of information is knowledge of the local data point density $\rho(\mathbf{x}_i)$ as described above. Here, for two different kernels the mislocation is analysed in 1-D using random samples.

Spline and boxcar kernels

First, two different averaging kernels are introduced that satisfy the zeroth and first order moment constraints (6.2) and (6.3). In practical applications the kernel must also have a finite support. The most important attribute of the shape of a kernel, with respect to mislocation, is its smoothness.

As an example of a smooth kernel is used the cubic spline $K_s(x)$, a piecewise third order polynomial:

$$K_s(x) = \begin{cases} \frac{2}{3} - |x|^2 + \frac{1}{2}|x|^3 & \text{for } |x| \leq 1 \\ \frac{4}{3} - 2|x| + |x|^2 - \frac{1}{6}|x|^3 & \text{for } 1 < |x| \leq 2 \\ 0 & \text{for } |x| > 2. \end{cases} \quad (6.17)$$

The boxcar kernel $K_b(x)$, serves as an example of a non-smooth kernel:

$$K_b(x) = \begin{cases} \frac{1}{2} & \text{for } |x| \leq 1 \\ 0 & \text{for } |x| > 1. \end{cases} \quad (6.18)$$

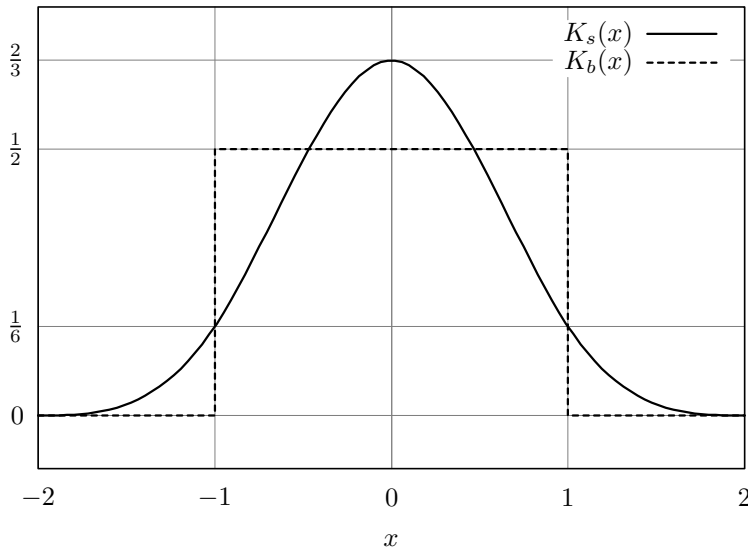


Figure 6.1: Cubic spline averaging kernel $K_s(x)$ (6.17) and boxcar averaging kernel $K_b(x)$ (6.18). The kernels have a different support, but have an equal width according to the second order moment criterion (6.19).

Both kernels are displayed in Figure 6.1.

According to their respective definitions, the spline and boxcar kernels have different regions of support. They have been defined in such a way, that they have equal width according to their second order moment:

$$\int_{-1}^1 K_b(x)x^2 dx = \int_{-2}^2 K_s(x)x^2 dx. \quad (6.19)$$

The second order moment is a plausible measure for the width of an averaging kernel, because it stands for the amount of smoothing it applies to second order polynomials. Two kernels of the same width in terms of second order moment therefore have an equal response up to second order.

Convergence of mislocation for random data points

Figure 6.2 shows the result of an analysis of mislocation as a function of the number of randomly distributed data points within the support of a 1-D kernel. It shows that knowledge of the local data point density – calculated as half the space between the first neighbouring data points to the left and to the right – is crucial. Without this knowledge the convergence is proportional to the square root of the number of data points and is independent of the shape of the kernel. If the data

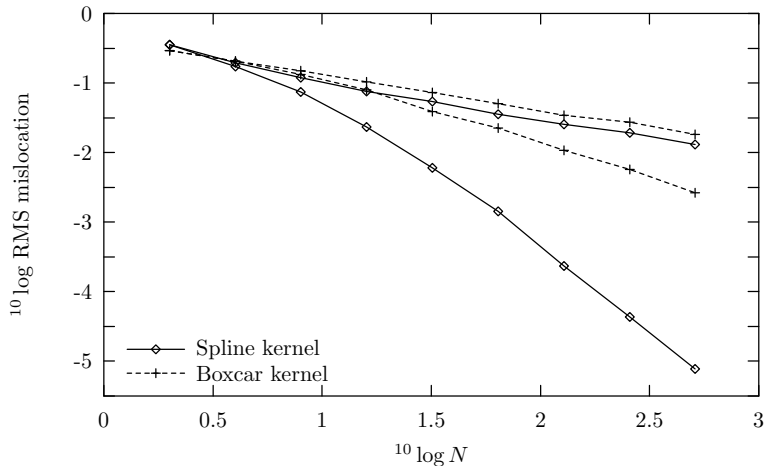


Figure 6.2: Convergence of mislocation as a function of the number N of randomly distributed data points within the support of the kernel. Each data point represent the root-mean-square (RMS) average of 100 random data point distributions. Two curves are shown for both the spline and the boxcar kernels. The lower curve is obtained with knowledge of the local data point density ρ , the upper without. The upper curves for both kernels show an asymptotic gradient of roughly -0.5 , the lower curves show asymptotic gradients of roughly -1.0 and -2.5 respectively. For explanation see Section 6.2.2.

point density is known, however, the smooth spline kernel shows a much faster convergence than the discontinuous boxcar kernel.

6.2.3 Regularisation with enhanced accuracy

Higher order moment constraints

As in the case of interpolation, our interest is to enhance the accuracy of the regularisation technique discussed above. The first order accuracy of averaging integral (6.5) is due to the constraints (6.6) and (6.7) on kernel $K(\mathbf{x})$. A straightforward approach to enhancing the accuracy of the averaging integral is to increase the number of constraints on $K(\mathbf{x})$. The extra constraints will take the form of moment constraints, such as for the k -th order moment:

$$\iint_{\text{supp}(K)} K(\boldsymbol{\xi})(\mathbf{x} - \boldsymbol{\xi})^{[k]} d\boldsymbol{\xi} = \mathbf{x}^{[k]}, \quad (6.20)$$

where superscript $[k]$ is again the iterated tensor product as defined in Equation (4.23). Note that this general form is compatible with the zeroth and first order moment constraints (6.6) and (6.7).

Hence, in order to obtain an n -th order accurate averaging integral, one may design a kernel $K(\mathbf{x})$ that satisfies constraint (6.20) for $k = 0 \dots n$. Although this makes sense in the continuous formulation, it is not very useful in practice, because in the discrete case it is difficult to satisfy the moment constraints for arbitrary data points. The mislocation discussed above, for example, is the result of not being able to satisfy already the first order moment constraint.

Averaging extrapolations

A more practical method for enhancing the accuracy of the regularisation is available if the data points also contain derivative information. In that case extrapolations from a number of data points can be averaged. The normal averaging of the data can then be seen as the averaging of zeroth order extrapolations. As in Chapter 4 both the Taylor expansion and the Dutch Taylor expansion can be used for the extrapolation. In theory, the Dutch Taylor expansion again promises an accuracy of one order higher than an individual Taylor expansion for an averaging kernel that is exact to first order. This first order accuracy, however, is not obtained in practice due to the mislocation discussed above.

To assess the combined effects of mislocation and the averaging of extrapolations an experiment is performed on the approximation of cosines. The experiment is similar to that discussed in Section 6.2.2, but now for cosines rather than linear functions. The cosines have a wavelength of $\pi/4$ times the support of a spline kernel, and a random phase. Every experiment uses the spline kernel and the estimated local data point density, as described above. Averaging integrals are calculated using both the normal and the Dutch Taylor expansions up to second order. The Dutch Taylor expansion for $n = 0$ is equal the Taylor expansion for $n = 0$ (see also Chapter 4).

Clearly, the accuracy of averaging the Dutch Taylor expansion depends strongly on the mislocation. For sufficiently high N , these expansions lead to a greater accuracy than the corresponding Taylor expansions, but in practical applications the number of data points will be low. Hence, the Dutch Taylor expansion is not as useful in regularisation based on an averaging integral as it is interpolation. The reason is that the averaging integral cannot be made accurate to first order in practice. In the following only the Taylor expansion will be used.

6.3 Paraxial ray tracing

6.3.1 Ray field mapping by averaging

The goal of the paraxial ray method described in this section is to construct a ray field map expressing the ray field coordinates $\boldsymbol{\xi} \equiv (\tilde{\mathbf{x}}_0, T, \tilde{\boldsymbol{\phi}}_0)$ as a function of the position/angle coordinates $\mathbf{y} \equiv (\mathbf{x}, \boldsymbol{\phi})$, as discussed in Chapter 3.

The algorithm basically has two tasks. The first task is to solve the ray equations (Chapter 2) to provide $\mathbf{y}(\boldsymbol{\xi})$. The second task is to determine for each grid

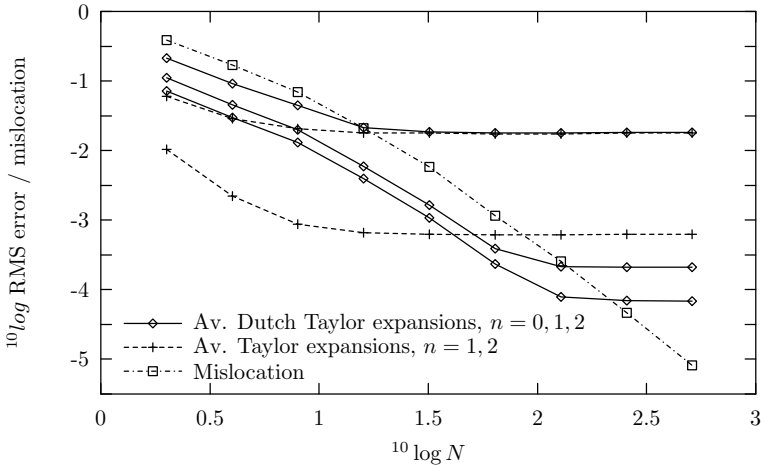


Figure 6.3: Convergence of the approximation of cosines as a function of the number N of randomly distributed data points within the support of the kernel. Each data point represents the root-mean-square (RMS) average of 100 random data point distributions. All curves use spline kernels and estimates of local data point density. For further explanation see Section 6.2.3.

point \mathbf{y}^* the $\boldsymbol{\xi} = \boldsymbol{\xi}^*$ that solves

$$\mathbf{y}(\boldsymbol{\xi}) = \mathbf{y}^*. \quad (6.21)$$

In the ray field construction method (Chapter 5) this second task is performed by determining a local linear approximation to $\mathbf{y}(\boldsymbol{\xi})$, which is then solved by means of linear inverse interpolation.

In the paraxial ray method proposed here, system (6.21) is solved approximately using the averaging integrals discussed in the previous section. Formally, the solution $\boldsymbol{\xi}^*$ to (6.21) can be expressed as a representation integral:

$$\begin{aligned} \boldsymbol{\xi}^*(\mathbf{y}^*) &= \int \boldsymbol{\xi}(\mathbf{y}) \delta(\mathbf{y}^* - \mathbf{y}) d\mathbf{y} \\ &= \int \boldsymbol{\xi} \delta(\mathbf{y}^* - \mathbf{y}(\boldsymbol{\xi})) \det \left(\frac{\partial \mathbf{y}}{\partial \boldsymbol{\xi}} \right) d\boldsymbol{\xi}. \end{aligned} \quad (6.22)$$

Hence, $\boldsymbol{\xi}^*(\mathbf{y}^*)$ can be expressed as an integral over $\boldsymbol{\xi}$. For practical applications this integral must be adapted to provide an approximate solution:

$$\boldsymbol{\xi}^*(\mathbf{y}^*) \approx \int \boldsymbol{\xi} K(\mathbf{y}^* - \mathbf{y}(\boldsymbol{\xi})) \det \left(\frac{\partial \mathbf{y}}{\partial \boldsymbol{\xi}} \right) d\boldsymbol{\xi}, \quad (6.23)$$

with K an averaging kernel. The order of accuracy of (6.23) depends on the properties K , as discussed in Section 6.2).

A simple implementation of a paraxial ray tracing algorithm works by shooting rays from a regular grid in $(\tilde{\mathbf{x}}_0, \tilde{\phi}_0)$ with regular steps of T into the medium. The evaluation points of the ray field then form a regular grid in $\boldsymbol{\xi}$, but an irregular distribution in \mathbf{y} . Integral (6.23) for grid point \mathbf{y}^* can be evaluated approximately using the data points that fall within the support of K . The jacobian $\det(\partial\mathbf{y}/\partial\boldsymbol{\xi})$ then serves as the estimator of the local data point density ρ mentioned in Section 6.2.

The defining characteristic of paraxial ray methods is the use of paraxial derivatives to extrapolate information from the rays in the ray field to the grid points in the mapping domain. Rather than averaging the ray field coordinates $\boldsymbol{\xi}$ to find $\boldsymbol{\xi}^*(\mathbf{y}^*)$ in (6.23) it is more accurate to average paraxial estimates to the latter from each of the data points. This can be done using the Taylor expansion (4.22):

$$\boldsymbol{\xi}^*(\mathbf{y}^*) \approx \int \mathcal{T}_n[\boldsymbol{\xi}](\mathbf{y}^*; \mathbf{y}) K(\mathbf{y}^* - \mathbf{y}(\boldsymbol{\xi})) \det\left(\frac{\partial\mathbf{y}}{\partial\boldsymbol{\xi}}\right) d\boldsymbol{\xi}. \quad (6.24)$$

In practice only first order paraxial derivatives are calculated:

$$\mathcal{T}_1[\boldsymbol{\xi}](\mathbf{y}^*; \mathbf{y}) = \boldsymbol{\xi}(\mathbf{y}) + (\mathbf{y}^* - \mathbf{y}) \cdot \frac{\partial\boldsymbol{\xi}}{\partial\mathbf{y}}. \quad (6.25)$$

These paraxial derivatives must be calculated along the rays as explained in the following section.

6.3.2 Paraxial ray information

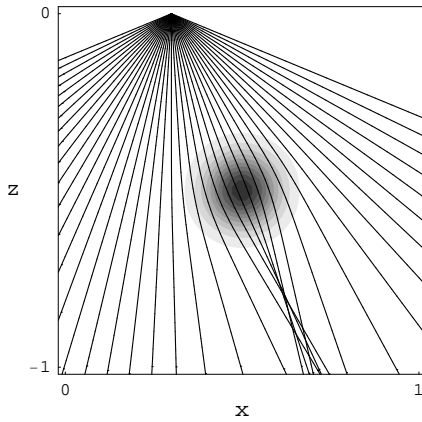
The derivatives $\partial\boldsymbol{\xi}/\partial\mathbf{y}$ of Equation (6.25), i.e., the partial derivatives of the position/angle coordinates with respect to the ray field coordinates, can easily be calculated along the rays with the help of the theory of Chapters 2 and 3. The derivatives can be identified as the inverse of a set of paraxial derivatives:

$$\frac{\partial(\tilde{\mathbf{x}}_0, T, \tilde{\phi}_0)}{\partial(\mathbf{x}, \phi)} = \left(\frac{\partial(\mathbf{x}, \phi)}{\partial(\tilde{\mathbf{x}}_0, T, \tilde{\phi}_0)} \right)^{-1}. \quad (6.26)$$

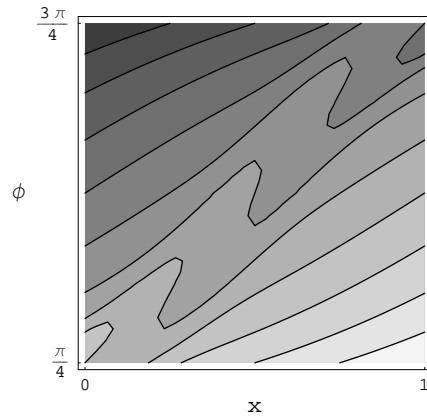
The paraxial derivatives can be calculated directly along the ray using the paraxial ray equations (2.36). However, inverting this paraxial derivative matrix is not a good idea. The shear-like ray field deformations observed in the position/angle domain may make this derivative matrix nearly degenerate, which, in turn, may lead to considerable numerical errors in the matrix inversion.

Instead, it is better to use a formulation in terms of the propagator matrix $\mathbf{P} = \partial(\mathbf{x}, \mathbf{p})/\partial(\mathbf{x}_0, \mathbf{p}_0)$, which, as shown in Equation (2.42), is easily invertible:

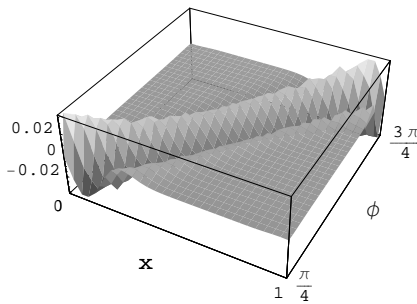
$$\left. \frac{\partial(\tilde{\mathbf{x}}_0, T, \tilde{\phi}_0)}{\partial(\mathbf{x}, \phi)} \right|_{\mathcal{H}} = \left. \frac{\partial(\tilde{\mathbf{x}}_0, T, \tilde{\phi}_0)}{\partial(\mathbf{x}_0, \mathbf{p}_0)} \right|_{\mathcal{H}} \cdot \mathbf{P}^{-1} \cdot \left. \frac{\partial(\mathbf{x}, \mathbf{p})}{\partial(\mathbf{x}, \phi)} \right|_{\mathcal{H}}. \quad (6.27)$$



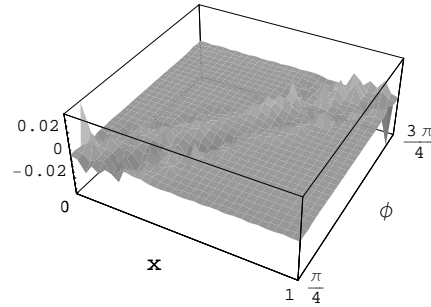
(a) Medium used in paraxial ray tracing example and rays shot for one source location



(b) Contours of \tilde{x}_0 evaluated at $z = -1$; contour in the middle represents $\tilde{x}_0 = 0.5$, other contours in steps of 0.25



(c) Error in \tilde{x}_0 for paraxial ray method with averaging integral



(d) Error in \tilde{x}_0 for paraxial ray method using only nearest ray point

Figure 6.4: Example of the use of paraxial ray tracing in the position/angle domain. For explanation see Section 6.3.3.

Although this formulation requires knowledge of the full ray propagator \mathbf{P} it does not require the solution of a higher number of paraxial ray equations. Only four columns of \mathbf{P} need to be calculated explicitly, because the other two can be obtained from the analytical solutions (2.43) and (2.44). The two other derivative matrices on the right hand side of (6.27) can be calculated using (3.19), (3.20) and (3.24).

6.3.3 Example

As an example of the paraxial ray tracing method in the position/angle domain a ray field map is constructed for an isotropic medium with a Gaussian-shaped slow anomaly on top of a homogeneous background. A contour plot for the medium is shown in Figure 6.4(a). Phase velocity V satisfies $V^{-1} = 1.0 + 0.3 \exp(-100.r^2)$, with r the distance to the point $(0.5, 0.5)$.

Rays are shot downwards from the surface $z = 0$, parameterised by \tilde{x}_0 , $\tilde{\phi}_0$, and T . The rays are calculated using a Runge-Kutta scheme as in Chapter 5 with a time step of 0.025. The ray field map is constructed on a rectangular grid in (x, z, ϕ) -space, with 31 grid points for $x \in [0, 1]$ and $z \in [-1, 0]$, and 31 grid points for $\phi \in [\pi/4, 3\pi/4]$. The rays shot from a single source position are drawn in Figure 6.4(a). The 41 rays cover a range of $\tilde{\phi}_0 \in [1\pi/8, 7\pi/8]$. The shot positions also cover a wider range than the map in order to obtain arrivals at depth $z = -1$ for all angles. The 123 source positions cover a range of $\tilde{x}_0 \in [-1, 2]$.

For comparison a reference calculation is made by tracing rays up towards the surface for every grid point at $z = -1$. The resulting contour plot for \tilde{x}_0 is shown in Figure 6.4(b). The folding of contours over the x -range indicates the presence of multi-pathing; the turning points correspond to the caustics.

For the averaging integral (6.24) a spline kernel (6.17) is used with a support of exactly 4 grid intervals in all three directions. The result of the paraxial method with the averaging kernel is evaluated in Figure 6.4(c). It shows the error in the calculation of x_0 with respect to the reference calculation shown in Figure 6.4(b). The errors are largest near the caustics, which is the result of the smoothing caused by the averaging. The overall error is reasonably small, and may be reduced by choosing smaller support for the averaging kernels and increasing the number of rays traced.

For comparison Figure 6.4(d) shows an alternative approach to paraxial ray tracing. At each grid point only one paraxial ray estimate is evaluated from the ray point that is nearest to the grid point. An advantage is that the smoothing is reduced with respect to 6.4(c). A disadvantage is that the errors are more or less randomly distributed. This is a problem for further application of the ray field maps, because this type of error is harmful for the accuracy of finite difference estimates of derivatives.

6.4 Discussion and conclusions

It has been shown that ray field maps in the position/angle domain can be calculated using paraxial ray methods. Rays are traced downwards from the acquisition surface, adding ray field information to those grid points that are contained in a paraxial neighbourhood of the ray. The algorithm is very simple and easy to implement for smooth media of arbitrary complexity; the only ingredients needed are a model description (e.g., grid and interpolator) a data structure for the grid

in the position/angle domain and a paraxial ray tracer.

The algorithm is also very efficient. Because it yields the ray field information organised by angles at depth, it must be compared to other algorithms that yield the same. Currently the only possibility considered by studies regarding imaging in the angle domain seems to be to simply trace the rays upwards from the subsurface image points (e.g., Koren et al., 2002; Brandsberg-Dahl et al., 2003). Obviously, tracing upwards is much more expensive, because the image points extend over one dimension more than the acquisition points. Use of the proposed paraxial ray method instead leads to a gain in speed for the ray tracing roughly proportional to the average length of the rays divided by spatial interval of the image grid.

Note that the paraxial ray algorithm is perfectly applicable to a target-oriented approach as described in Koren et al. (2002). From the boundary of the target rays may first be shot outwards to the acquisition surface to determine the ray field coordinates for each ray. These coordinates may then be propagated inside the target by continuing the rays inward from the boundary.

A general advantage of tracing ray fields in the position/angle domain is that the geometrical spreading, and hence the ray density, does not depend on the length of propagation or some other aspect of the ray path. Apart from the ray density at the source and the specific parameterisation of the slowness surface the ray density only depends on the local phase velocities (see Section 3.4). Hence, the ray density at depth can be predicted reasonably well and the number of rays shot from the acquisition surface can be chosen a priori in such a way that a sufficient number of rays passes in the neighbourhood of each grid point.

An interesting point to note here concerns the occurrence of shadow zones for ray tracing in the spatial domain. Although perfect shadow zones do not exist in smooth media, the geometrical spreading may become very high at places. In wave front construction codes this leads to a large number of ray insertions to maintain a certain accuracy. For ray tracing in the position/angle domain, however, these zones do not have the same effect. The fact that the local ray density in the position/angle domain primarily depends on the local phase velocity promises that a zone that is “left open” by the rays from one source location will be filled in by rays from neighbouring source locations.

The above observation may have very interesting consequences. It seems that the number of rays required to calculate a single ray field map in the position/angle domain can be much smaller than that required for the calculation of a series of ray field maps in the spatial domain for the same range of source positions. If this is true, it may very well lead to the conclusion that even if (possibly multi-valued) ray field maps in the spatial domain are desired, it is advantageous to first construct a ray field map in the position/angle domain, followed by the extraction of spatial ray field maps for the individual source locations. This is left as another area for future research.

The paraxial ray method developed so far is shown to work adequately in the example of Section 6.3.3. A number of issues have not been addressed, however, such as the choice of grid spacing to be used and the number of rays to be shot

and the effect on the accuracy. Although the geometrical spreading in the position/angle domain is independent of the length of propagation, the same does not hold for the number of rays required to maintain sufficient accuracy. In general, the variations in both ray field and ray field map will become smaller in scale along the way. Hence, the number of rays and grid points required to approximate these variations will in general increase during propagation.

An improved version of paraxial ray method for the position/angle domain will therefore probably contain a mechanism to increase the number of rays along the way. It does not have to be as complex, however, as the mechanism used in wave front construction methods (e.g., Chapter 5). It is probably sufficient to use statistical rather than deterministic criteria for the addition of rays, based on the propagation distance and the length scales and sizes of variations in the medium. The ray field may, for example, be re-initialised with enhanced ray density at intermediate surfaces at depth. These criteria should preferably be determined rigourously from experiments on suites of random models, as suggested also in the discussion of Chapter 5.

