# Stochastic Local Search for BRDF Parameterization

Jonathan Backer, Fred Kimberley, and Iryna Skrypnyk

University of British Columbia

Department of Computer Science

{backer, fkimber, skrypnyk}@cs.ubc.ca

**Abstract**

A bidirectional reflectance distribution function (BRDF) models how light is reflected by a surface. There is an extensive literature [8] advocating different BRDF models on the basis of computational complexity, expressiveness, theoretical models, and physical correctness. We examine the parameterization of the Lafortune BRDF model [4], a generalization of an empirical model commonly used in rendering, from sample data using stochastic local search techniques.

## 1 Background

A BRDF is the ratio of incoming to outgoing light intensity at a particular point on a surface. Denoted $R_\lambda(P, U, V)$, it is a function of wavelength $\lambda$, surface point $P$, incident light direction $U$, and exitant light direction $V$. $R_\lambda(P, U, V)$ is typically positive: $R_\lambda(P, U, V) \geq 0$, energy conserving: $\int R_\lambda(P, U, V)\, dV \leq 1$, and symmetric: $R_\lambda(P, U, V) = R_\lambda(P, V, U)$. Algorithms, such as stochastic ray-tracing, may depend on these properties.
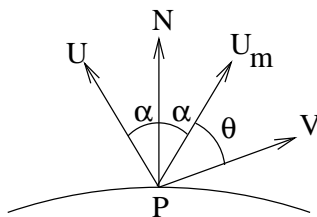


Figure 1: Angles and unit vectors for BRDF models.

Completely diffuse surfaces, such as a blackboard, scatter light uniformly in all directions and $R_\lambda = \rho_d C_d$, where $\rho_d \in [0, 1]$ is the maximum intensity of reflected light and $C_d$ is a normalization factor guaranteeing energy conservation. Completely reflective surfaces, such as mirrors, are represented with $R_\lambda = \lim_{n \to \infty} \rho_s (U_m \cdot V)^n = \lim_{n \to \infty} \rho_s \cos^n \theta$, where $\rho_s \in [0, 1]$ is the maximum intensity of reflected light. Phong [6] first proposed using $R_\lambda = \rho_d C_d + \rho_s C_s \cos^n \theta$ to model isotropic surfaces, i.e. surfaces with reflection invariance with respect to rotation about the surface normal. The second term in the equation is called a *cosine lobe* and captures the

specularity of a surface, which is parameterized by $n \in [1, \infty)$. Analogous to the diffuse model, $\rho_s \in [0, 1]$ is the maximum intensity of specular lobe and $C_s$ is a normalization constant ensuring energy conservation. Typically, $\rho_d$ and $\rho_s$ are selected such that $\rho_d + \rho_s \leq 1$ to secure energy conservation.

Lafortune et al. [4] align the $z$-axis to the normal vector and the $x, y$-axes to the principal directions of anisotropy, if the surface is anisotropic. After this change of coordinates, $C_s$ is eliminated and the dot product of the specular term $U_m \cdot V$ is replaced with a clamped weighted dot product $\max\{0, C_x U_x V_x + C_y U_y V_y + C_z U_z V_z\}$. The Phong model is a special case of this generalization where $-C_x = -C_y = C_z = \sqrt[n]{C_s}$. Lafortune's generalized cosine lobe can reproduce effects such as off-specular reflection (reflection not centred around the mirror vector), retro-reflection (reflection in the direction of incidence), and anisotropy with the appropriate choice of weights. Moreover, these effects can be trivially combined with the inclusion of several specular lobes.

Most real BRDFs become very specular at grazing angles as demonstrated in Figure 2, but the Lafortune model actually becomes more diffuse because a substantial volume of the specular lobe is hidden below the surface. To compensate for this, Lafortune et al. direct the portion of the lobe hidden below the surface into a completely reflective component, which is scaled with a reflective coefficient $R_m \in [0, 1]$.



Figure 2: The table surface becomes more specular at grazing angles (image from [4], used without permission).

The original Phong model is positive, energy conserving, and symmetric. The Lafortune model is positive and symmetric but $C_x, C_y$, and $C_z$ must be carefully chosen to make the model energy conserving.

The BRDF model that we use lacks this additional reflective coefficient, scales the weights of the specular lobes to eliminate $\rho_s$, and uses a rigid rotation around the surface normal to align the $x, y$-axes to the principal directions of anisotropy. We constrain the parameters to the reasonable bounds in Table 1. The range for $C_x, C_y$, and $C_z$ is almost 50% larger than required for the Phong model and the range for specularity is about an order of magnitude larger than encountered in our experiments with real data.

In this paper, we examine the use of Stochastic Local Search techniques to find parameterizations of this BRDF model. A particular problem instance to be solved consists of a fixed number of cosine lobes and a data set of BRDF values with corresponding incoming vector and exitant vector pairs. The objective function that is minimized is the root mean square error (RMSE) over this data set.

| BRDF Parameter | Description | Lower Bound | Upper Bound |
| --- | --- | --- | --- |
| $\rho_s$ | reflectance factor | 0.0 | 1.0 |
| $n$ | specularity | 0.0 | 10000.0 |
| $C_x$ | weight | -1.5 | 1.5 |
| $C_y$ | weight | -1.5 | 1.5 |
| $C_z$ | weight | -1.5 | 1.5 |
| $pan$ | rotation about surface normal | $-90°$ | $90°$ |

Table 1: Lower and upper bounds for BRDF parameter values.

## 2 Algorithms

Three of our approaches leverage N2FB, a sophisticated Levenberg-Marquardt non-linear least-squares gradient descent optimization routine with simple parameter bounds. It is part of the Port 3 collection of numerical routines developed by Bell Labs and is freely available at `http://netlib.org` for non-commercial use. As a gradient descent algorithm, N2FB only guarantees local optimality and is very sensitive to where the search begins. We also examined the downhill simplex method as described in [7].

### 2.1 Adaptive Evaluation

In [1], we made three casual observations about gradient descent methods in general and N2FB in particular.

1. A gradient descent method may behave very differently given different data samples of equal size drawn from the same distribution because the sample data define the search landscape.

2. The runtime of a gradient descent method is proportional to the amount of sample data.

3. A minimum amount of sample data is typically required for meaningful convergence.

AED, Adaptive Evaluation with Decreasing subsample size, is a novel dynamic local search algorithm that leverages these observations. The key idea of AED is to subsample the original data set without replacement and apply N2FB on the subsample. We adaptively change the subsample size and repeat the process because the subsample may be unrepresentative or may be less than the critical amount for convergence.

An individual element in a small data subsample will have more influence on the search landscape than it would in a large subsample. Our intuition is that smaller subsamples will diversify the search by amplifying random errors in the sample data. Conversely, larger subsamples will intensify our search with a more accurate representation of the true landscape which limits the effects of individual errors.

We initially used the RMSE between the candidate solution and measured BRDF over the entire data set as a performance metric. After a gradient descent, if the RMSE decreased then we increased the subsample size, otherwise we eventually decreased the subsample size. The history mechanism of eventual decrease was necessary so as not to overreact and abandon a promising region of the search space.

Through experimentation we noticed that the error code returned by N2FB was a good indicator of search progress. That is, if N2FB returned an *iteration limit* then the candidate solution was promising and warranted further exploration. However, if N2FB returned a *singular convergence* or *false convergence* then the returned candidate solution usually had too many degrees of freedom or was pulled against the simple bounds of the search space. We simplified our algorithm with no apparent performance penalty by increasing the subsample size on an iteration limit and decreasing it on a singular or false convergence.

Finally, to start our adaptive search with a quality candidate solution we repeatedly randomly pick candidate solutions and perform just a few iterations of N2FB until N2FB returns an iteration limit. The resulting pseudo code is in Figure 3 where $\phi$ denotes the rate of change in the sample size and $\rho_0$ denotes the initial sample size.

## 2.2 Adaptive Evaluation Without Decreasing Subset Size

Preliminary testing suggested that AED frequently decreased $\rho$ in succession until it prompted a restart. So to study how decreasing the subsample size diversifies AED's search, a simpler variant of AED was implemented. The AERS algorithm preserves all of the features of AED, including the algorithm parameters, except for its treatment of N2FB's failure to converge. Whereas AED handles this case by decreasing the current subsample size, AERS simply restarts.

## 2.3 Iterated Local Search: Combining Simulated Annealing with Gradient Descent

Instead of introducing a random element to diversify a gradient descent algorithm, we can interleave its fast deterministic steps with a randomized search method, which is slower but less prone to getting stuck in local minima. With a good balance of gradient descent and randomized search steps, the hybrid algorithm will explore larger areas of the search space and escape from low quality local minima while maintaining good convergence time.

We chose a basic variant of a simulated annealing algorithm [5] as a randomized search mechanism. The main idea behind this method is to gradually reduce the size of its randomized search steps, or perturbations, such that the search eventually converges to the vicinity of a high quality local minimum. The probability of accepting a worsening search step proposed by the perturbation depends on both the quality of the proposed candidate solution and the temperature parameter $T$, which controls the degree of perturbation and is continually cooled during the search. The outline of SA, the simulated annealing method used in our iterated local search, is shown in Figure 4.

The temperature is cooled according to a simple geometric schedule, in which a current tempera-

**procedure** *AED* $(\pi, \mathcal{S}, e)$
    **input** *problem instance* $\pi \in \Pi$*, sample data* $\mathcal{S}$*, cutoff error* $e$
    **output** *candidate solution* $s$

*restart:*
    **do**
        $s := \text{RandomPick}(\pi)$
        $S := \text{Subsample}(\mathcal{S}, \rho_0)$
        $s := \text{N2FB}(\pi, S, s)$
    **until** (n2fb_err $\neq$ ITERATION_MAX)

    $\rho := \rho_0$
    **while** (RMSE$(\mathcal{S}, s) > e$) **do**
        $S := \text{Subsample}(\mathcal{S}, \rho)$
        $\hat{s} := \text{N2FB}(\pi, S, s)$
        **if** (n2fb_err = ITERATION_MAX) **then**
            $s := \hat{s}$
            **if** ($\rho = |\mathcal{S}|$) **then goto** *restart*
            $\rho := \rho \cdot \phi$
            **if** ($\rho \geq |\mathcal{S}|$) **then** $\rho = |\mathcal{S}|$
        **else**
            $\rho := \rho/\phi$
            **if** ($\rho < \rho_0$) **then goto** *restart*
        **end**
    **end**
    **return** $s$
**end** *AED*

Figure 3: Algorithmic outline of AED.

```
procedure SA(π', s, T_start, T_stop, N_t, N_i)
    input problem instance π' ∈ Π', objective function f(π), candidate solution s,
    start temperature T_start, stop temperature T_stop, number of temperatures N_t,
    number of iterations at each temperature N_i
    output candidate solution s'
    s' := s
    T := T_start
    for t from 1 to N_t do
        for i from 1 to N_i do
            s* := perturb(π', s', T)
            s' := accept(π', s*, s', T)
        end
        T := cT
    end
    return s'
end SA
```

Figure 4: Algorithmic outline of SA.

ture is multiplied by a constant factor [5]:

$$
\begin{aligned}
T &= cT \\
c &= \left( \frac{T_{stop}}{T_{start}} \right)^{1/(N_t - 1)}
\end{aligned}
$$

At each temperature, the current candidate solution $s$ undergoes a random perturbation, which is analogous to the shaking of the atoms of metal in physical annealing. The degree of perturbation is controlled by the current temperature, i.e. larger changes to $s$ are more likely at the beginning of the search. The perturbation results in a neighbouring candidate solution $s^*$, as shown in Figure 5. Since each parameter component $s_p$ of $s$ has range constraints, the lower and upper bounds of $s_p$ are taken into account in order to generate a valid solution. Note that $T$ parameterizes the amount of noise added to the current value of $s_p$, after being appropriately scaled by that parameter's range.

The criterion for accepting a perturbed candidate solution $s^*$ as a next search step should generally favour higher quality candidate solutions, while still allowing worsening steps at higher temperatures. This should result in a search trajectory that is more constrained to improving search steps as the temperature is reduced. To achieve this effect, a Metropolis acceptance criterion is used [2]:

$$
P_{accept}(T, s, s') = \begin{cases} 1 & : \quad f(s') < f(s) \\ \exp\left( \frac{f(s) - f(s')}{rT} \right) & : \quad otherwise \end{cases}
$$

where $r$ is a constant factor.

We studied the effects of combining SA with N2FB into a hybrid search algorithm SALS, which is outlined in Figure 6. The algorithm is initialized by randomly picking a starting candidate solution.

```
procedure perturb(π', s, T)
    input problem instance π' ∈ Π', candidate solution s, temperature T
    output candidate solution s*
    for p from 1 to N_p do
        s*_p := s_p + (ub_p − lb_p) × T × N(0, 1)
        if s*_p < lb_p
            s*_p := lb_p
        end
        if s*_p > ub_p
            s*_p := ub_p
        end
    end
    return s*
end perturb
```

Figure 5: Algorithmic outline of a perturbation mechanism.

Then SA is run for a few temperature iterations, after which a hybrid descent is performed by alternating N2FB with SA. The descent is terminated when a stagnation is detected, i.e. there no improvement in solution quality for a long time, and then the algorithm is restarted. The algorithm terminates when a solution of desirable quality is found or a time cutoff is exceeded.

The first time that SA is executed more temperature settings are used, and they start at higher values. The idea is to take advantage of the global nature of SA to find a promising area of the search space before starting a gradient descent. Running SA following the gradient descent at a few lower temperatures provides a chance for escaping from a local minimum in which the search may have become trapped.

## 2.4 Averaging: An Alternative Method for Combining Simulated Annealing with Gradient Descent

As with the previous algorithm, the goal of this algorithm is to combine the speed of gradient descent with simulated annealing's ability to escape from local minima. A hybrid may avoid many local minima and quickly escape from those that it encounters by using a well tuned simulated annealing algorithm and a good gradient descent method. We use SA for the simulated annealing and N2FB for the gradient descent; the former for its simplified mechanism and the later for its quick convergence and frequent ability to find a BRDF parameterization that was visually indistinguishable from the original in previous experiments.

The hybrid algorithm is initialized by randomly selecting a point in the search space. It then runs N2FB and SA independently until both reach some stopping criteria. Once both algorithms have both stopped their results are combined in an unweighted average. This average is then used as a new starting point for both algorithms and the process is repeated until either algorithm converges

**procedure** *SALS*$(\pi', T_{start}, T_{stop}, N_t, N_i)$

    **input** *problem instance* $\pi' \in \Pi'$*, objective function* $f(\pi)$,
    *temperature parameters* $T_{start}, T_{stop}, N_t, N_i$
    **output** *solution* $\hat{s} \in S(\pi')$
    $\hat{s} := RandomPick(\pi')$
    **while not** *terminate1*$(\pi', \hat{s})$ **do**
        $s := RandomPick(\pi')$
        $s := SA(\pi', s, T_{start}, T_{stop}, N_t, N_i)$
        **if** $f(s) < f(\hat{s})$
            $\hat{s} := s$
        **end**
        **while not** *terminate2*$(\pi', \hat{s}, s)$ **do**
            $s := N2FB(\pi', s)$
            **if** $f(s) < f(\hat{s})$
                $\hat{s} := s$
            **end**
            $s' := SA(\pi', s, T_{start}, T_{stop}, N_t, N_i)$
            **if** $f(s') < f(\hat{s})$
                $\hat{s} := s'$
            **end**
            $s := s'$
        **end**
    **end**
    **return** $\hat{s}$
**end** *SALS*

Figure 6: Algorithmic outline of SALS.

```
procedure AVG(π′)
    input problem instance π′ ∈ Π′, objective function f(π)
    output solution ŝ ∈ S(π′)
    while not terminate(π′, s) do
        s := RandomPick(π′)
        while not (stagnating(π′, s) or converged(π′, s)) do
            s′ := SA(π′, s)
            if f(s) < f(s′)
                ŝ := s
            end
            s″ := N2FB(π′, s)
            if f(s) < f(s″)
                s″ := s
            end
            ŝ := (s′ + s″)/2
        end
    end
    return ŝ
end AVG
```

Figure 7: Algorithmic outline of AVG.

on a minima that is good enough or stagnation is detected. There are many possible choices for the former stopping criterion. One possibility is that both points are within some distance $\epsilon$ of each other in the search space. Another alternative is that the objective functions of both points are close, regardless of the locations in the search space. To avoid stagnation a dynamic restart is added. If the value of the objective function does not change by a reasonable amount after two iterations then the algorithm restarts. See Figure 7 for the pseudo code for this algorithm.

## 2.5  Downhill Simplex Method

The Downhill Simplex method (DS) is a deterministic multidimensional optimization technique discussed in [7]. The DS method has the advantage that it does not require information about the derivative of the objective function to traverse the search space.

A simplex is a set $S$ of $N+1$ points and the convex hull that they define in a $N$-dimensional space $V$. For every $p \in S$ in a non-degenerate simplex, $S \setminus \{p\}$ is a basis for $V$.

DS initializes by placing one point randomly and then placing the other $N$ points by perturbing the first one. For each point the value of the objective function is calculated. The point with the highest objective function is reflected across the face defined by the other $N$ vertices. If the new point has a lower objective function than the old one, then it it replaces the old one. If the new point has an objective function lower than the lowest value of the objective function seen so far,

then the new point is moved twice as far from the plane as it was before. However, if the new point is worse than the second worst point in the simplex, then a contraction towards the face is attempted. If this contracted point still doesn't improve, then all of the points are moved closer to the point with the lowest objective function value.

The simplex method stops when a point reaches an objective function below a specified threshold or when the ratio between the worst point and the best point becomes to low. If this ratio becomes too low it indicates that the simplex may have contracted too far and further steps will only bring very small gains in the objective function value. In practice it is often useful to restart DS with one of the points in the new simplex being the best point found so far in order to avoid false convergence.

# 3  Experiments and Evaluation

All of our experiments were executed on 1 GHz Pentium III computers running Red Hat Linux 7.2. This is essential because there is no obvious notion of run length that can easily be compared between the different algorithms, although in retrospect the number of BRDF evaluations seems like a natural choice. The performance of each algorithm on a particular instance is captured as a runtime distribution (RTD) which represents the probability that the algorithm terminates in a fixed time $t$ with a solution with less than a fixed RMSE $q$. We use RMSE as a solution quality metric because this is the evaluation function being minimized and it is a function of the particular BRDF independent of the specific parameterization. The latter point is important because in the Lafortune model two different parameterizations can represent the same BRDF.

Our problem instances are either synthetic or from real world measurements with a gonioreflectometer, a device used to measure BRDFs directly. The synthetic instances were generated by selecting a Lafortune model BRDF, generating sample data by randomly sampling the BRDF over the hemispheres of incident and exitant directions with uniform probability, and adding Gaussian noise to the BRDF samples. The key advantage of this benchmark is that the we know that the data can be fit with the Lafortune model. Two major disadvantages are that the sampling distribution and noise model may be unrepresentative of real world problems.

We chose our threshold error quality $q$ by running AED for a considerable amount of time on each problem instance, noting the minimum error found, and setting the threshold error to be slightly above the minimum — typically by a few thousandths and certainly by no more than a hundredth. In practice this procedure leads to a quality cutoff corresponding to visual indiscernibility with respect to synthetic data sets. That is, when we examine traces of trial runs with synthetic data we observe that the RMSEs have multi-modal distributions. Subsequent rendering of the BRDFs indicates that these modes correspond to how many specular lobes were fit to the data. With our synthetic test cases all cosine lobes are quite visible, so a certain RMSE threshold must be crossed before the results are visually indiscernible from the true BRDF. So at least with regard to the synthetic data our choice of arbitrary error cutoff seems appropriate.

In our testing we included two other algorithms as a baseline for comparison: RPLS and ASA. RPLS is a naive random picking followed by N2FB which is repeated until a good so-

lution is found. Adaptive Simulated Annealing [3] is a form of simulated annealing available at `http://netlib.org`. Two notable features of ASA are its reannealing schedule and the use of the first derivative of the evaluation function to scale how parameters are perturbed.

## 3.1 Tuning

We tuned each algorithm to a simpler problem instance and hoped that the tuning generalized because it was computationally infeasible to tune each algorithm to each problem instance. Although our subsequent testing was no longer a measure of peak performance, it inadvertently measured robustness. Figure 8 is a rendering of the tuning BRDF and Table 2 is its Lafortune BRDF representation.
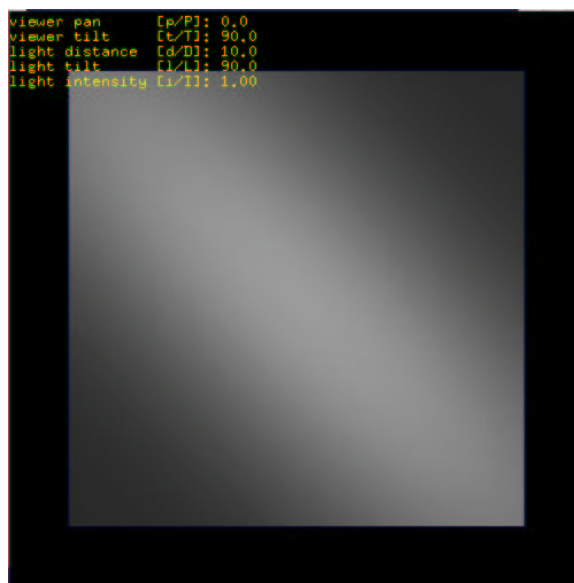


Figure 8: A single point light source illuminating a flat surface with the tuning BRDF.

| Lobe | $C_x$ | $C_y$ | $C_z$ | pan | n |
|---|---|---|---|---|---|
| I | -1.0 | 0.8 | 0.96 | 45.0° | 20.0 |
| Diffuse | 0.15 | | | | |

Table 2: Parameters for Lafortune BRDF used for tuning.

The incident and exitant hemispheres were uniform randomly sampled 10000 times to generate the tuning data. Gaussian noise with mean of 0 and a standard deviation of 0.1 was added to the BRDF calculations to roughen the search landscape. The RMSE acceptance threshold $q$ for this sample data was set to 0.11.

### 3.1.1 AED

To tune AED we iteratively adjusted $\phi$ and $\rho_0$ until we found a local minimum of $\phi = 1.400$ and $\rho_0 = 25$. The optimality of this minimum is demonstrated in Figure 9 by varying one parameter of this minimum while holding the other constant. Each of the RTDs in Figure 9 were generated with at least 700 independent trials.
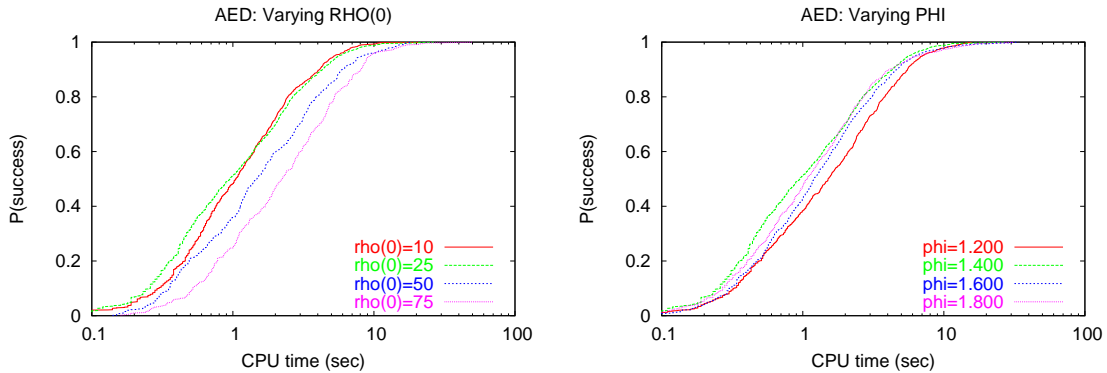


Figure 9: Tuning $\rho_0$ and $\phi$ for AED.

### 3.1.2 AERS

AERS was tuned analogously to find an optimal parameterization of $\phi = 1.0$ and $\rho_0 = 25$. Figure 10 shows the RTDs generated by 500 independent runs for various values of $\rho_0$ and $\phi$.
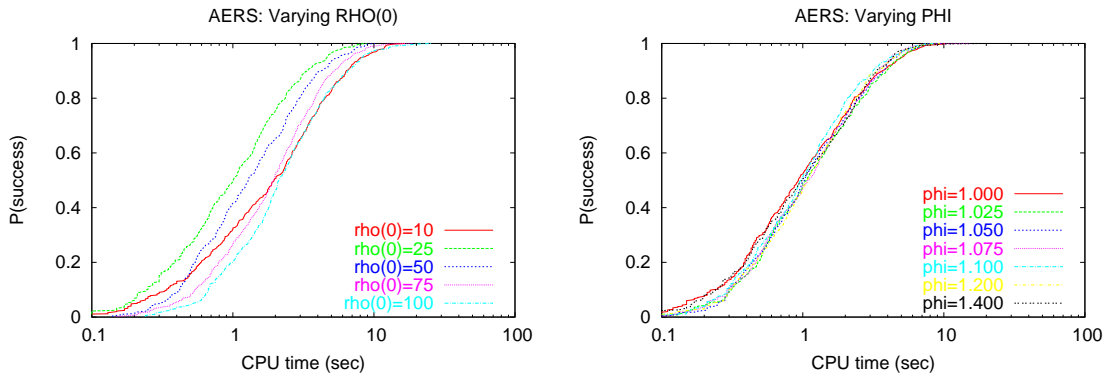


Figure 10: Tuning $\rho_0$ (left) and $\phi$ (right) for AERS.

Notice that varying $\phi$ has virtually no effect on the performance of the algorithm. Furthermore, setting $\phi$ to 1.0, that is, not increasing the subsample size following a successful gradient descent, produced roughly the same results as increasing the subsample size. This phenomenon suggests that there is a threshold subsample size sufficient for eventual successful convergence of the al-

gorithm, provided each subsample is random and restarts are performed often enough to diversify the search. Moreover, a reasonably small subset size allows for a fast execution of both N2FB, thus leading to the overall fast execution of AERS.

As a consequence of the parameter tuning, $\phi$ was eliminated from AERS in the subsequent experiments. So the key feature distinguishing AERS from RPLS is the fact that the former performs random picking followed by N2FB on a random subsample of data, as opposed to the full data set in the latter.

### 3.1.3  ASA

ASA has dozens of tunable parameters. We tweaked 4 parameters that previous experience and documentation suggested would have the most effect. We started with Temperature Ratio Scale which controls the cooling rate and final temperature because the documentation called it the most influential parameter. As the log of this value is used in actual computations we tried a wide range of values around the default of $10^{-5}$. The results in Figure 11 suggest that an optimal value lies between $10^{-4}$ and $10^{-5}$. Smaller values suffer from stagnation and larger values are probabilistically dominated. We split the difference and chose $5 \times 10^{-5}$ as our basis for further tuning.

Another value that directly influences the annealing schedule is Cost Parameter Scale Ratio which has a default value of 1.0. In Figure 11 a value of 0.9 clearly decreases stagnation.

The last two parameters that we tuned were Acceptance Frequency Modulus (AFM) and Generated Frequency Modulus (GFM) which determine the frequency of periodic testing and reannealing based on the number of accepted and generated states respectively. Prior experience suggested that AFM was the most influential of the two parameters so we tuned that first. Figure 11 shows that lower settings of AFM are slower to converge but suffer less stagnation, so we chose 25 as our setting which is substantially lower than the default of 100. GFM appears to have less influence so we chose 5000 which is half of the default of 10000. The result of our tuning was to reduce the expected runtime from $34.3 \pm 2.0$ standard deviations CPU seconds to $24.6 \pm 0.5$ standard deviations CPU seconds, a 28% decrease with a high confidence of decrease. However, there was practically no change in the median runtimes.

### 3.1.4  SALS

| Lobe | $C_x$ | $C_y$ | $C_z$ | pan | n |
|---|---|---|---|---|---|
| I | -1.0 | 0.8 | 0.96 | $45.0°$ | 20.0 |
| II | -1.0 | 0.8 | 0.96 | $-45.0°$ | 20.0 |
| Diffuse | 0.15 | | | | |

Table 3: Lafortune BRDF parameters for a cross brushed metal surface.

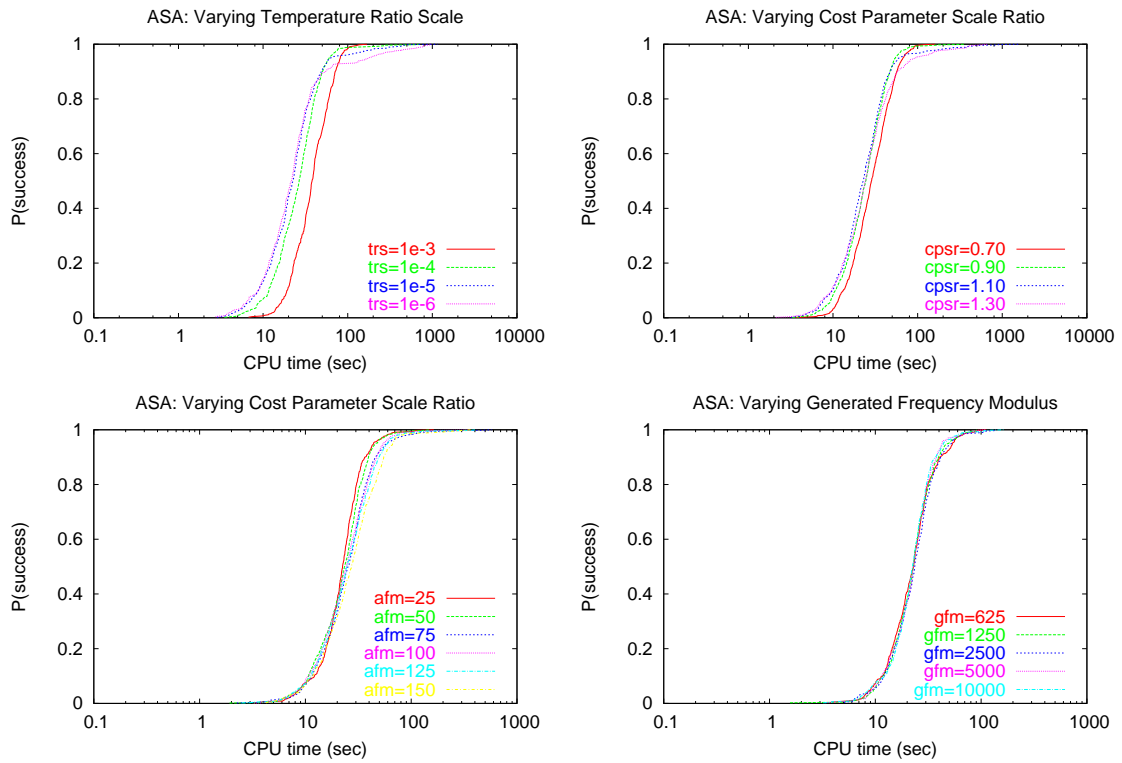The SALS algorithm was tested on 10000 tuples of synthetic data generated by the Lafortune

71

Figure 11: Tuning key ASA parameters. Each RTD is generated by more than 700 independent runs.
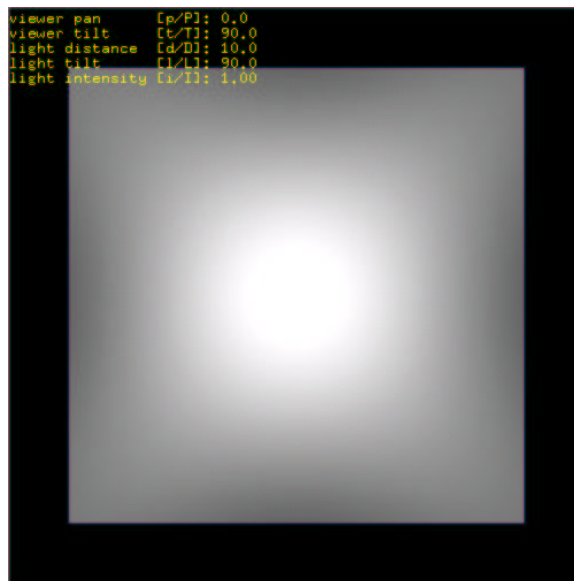


Figure 12: A single point light source illuminating a flat surface with a BRDF representing a cross brushed metal surface.

parameterization in Table 3 and Figure 12, with zero noise and Gaussian noise of 0.1 standard deviations. The threshold solution quality was a RMSE error of 0.02 for the noiseless case and 0.1 for the noisy case. A parameterization with 2 specular lobes was chosen, as it represents a more challenging problem instance to be solved by a pure gradient descent. By introducing a randomized SA search into the optimization process, we hoped to remedy the problem of finding a parameterization for a single lobe but not both — a result of getting stuck in poor local minima.

Most of the parameter tuning efforts for SALS were applied to finding effective starting and stopping temperature values for both variants of SA used in the algorithm. Another parameter that had to be adjusted was the number of iterations performed at each temperature. Table 4 shows the range of parameter values used in the course of the experiments. Over 50 combinations of parameter values were selected randomly from each range, such that $T_{start} > T_{stop}$.

| SA Parameter | Lowest Value | Highest Value |
|---|---|---|
| $T_{start}$ | 0.01 | 1.0 |
| $T_{stop}$ | 0.005 | 0.3 |
| $N_t$ | 1 | 50 |
| $N_i$ | 40 | 200 |

Table 4: Ranges of SA parameter values used for tuning.

Unfortunately, SALS performed poorly; unlike the other algorithms discussed in this paper, in all of the experiments SALS was not able to converge to the desired error tolerance, within the cutoff CPU time of 20 minutes. It was found that adjusting the temperature parameter values had little effect on improving the performance of the algorithm. In order to simplify parameter tuning, the Metropolis acceptance criterion was replaced by a constant acceptance ratio $A$, ranging between 0.05 and 0.2. Prior to descent, SA was not successful in finding a better starting search point than a simple random picking initialization. During descent, SA perturbations were not able to lead the search out of local minima. Instead, even at low temperatures significant variations of the error were observed during perturbations. Reducing the temperature even further usually resulted in the subsequent gradient descent ending up in the same local minimum from which an escape was attempted. A typical performance of SALS on noiseless synthetic data during hybrid descent is shown in Figure 13. Note the high error peaks during SA perturbations in between N2FB descents, and the overall stagnant behaviour of the algorithm.

The above observations led to a number of conclusions. It was initially thought that a problematic performance of the algorithm was due to its critical dependence on a good temperature selection, as well as the tight coupling of the perturbation mechanism and acceptance ratio, based on their shared use of the same current temperature value. However, replacing the temperature-dependent Metropolis acceptance criterion by a constant ratio and testing a wide range of temperature parameter combinations demonstrated that the algorithm's troubles are more deeply rooted. By closely examining the SA method and the problem domain itself, we have come to a conclusion that the algorithm's poor performance may be due to the fact that its perturbation mechanism is too primitive. The perturbation treats each BRDF parameter equally, scaling the temperature according to that parameter's allowable range. However, the range constraints of the parameters, imposed by the physical characteristics of the problem domain, are drastically different (see Table 1). To
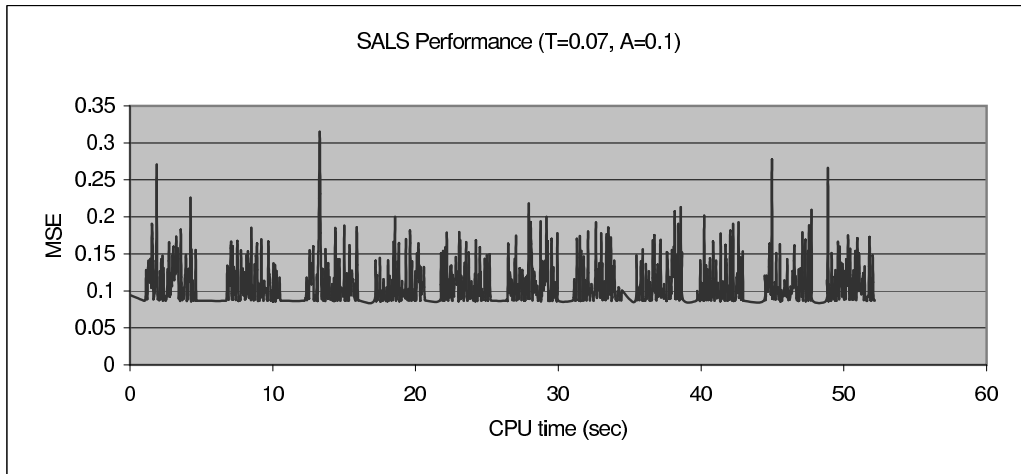
73

Figure 13: RMSE values measured during SALS execution.

complicate matters further, the RMSE is most sensitive to the exponential specularity parameter, which can take a value from 0 to 10000 – a range much larger than that of other parameters. It is likely that the failure of perturbations to steer the search in promising directions may be a result of this parameter imbalance.

### 3.1.5   AVG

The stopping criteria chosen for this method is when either SA or N2FB achieves a RMSE less than the threshold.

Both search routines were run until both found a local minima. Some trials were conducted where SA and N2FB were run for a set number of steps each and then averaged, however these changes did not improve the performance of the algorithm. The parameter settings for the SA algorithm can be found in the Highest Value column in Table 4.

In practice this algorithm performed extremely poorly; it was not able to converge within 1000 seconds of CPU time in any trial. Typical RMSE values for the single specular lobe tuning data set were just above 0.13 at the time cutoff, whereas the threshold was 0.11.

After the first few iterations, the SA algorithm moved extremely slowly compared to N2FB. As a result, N2FB would continually find a local minima and then after averaging the algorithm would be restarted at a point very close to where it had started the previous iteration. As a result, unless N2FB was lucky and was able to find a minima within the error bounds while SA's temperature was still high, the hybrid algorithm would fail to converge. For a detailed discussion of why SA traverses the search space so slowly see Section 3.1.4.

There are many modifications to this method that have yet to be tried such as different choices for the two local search algorithms, a weighted average of the returned solutions, and different criteria
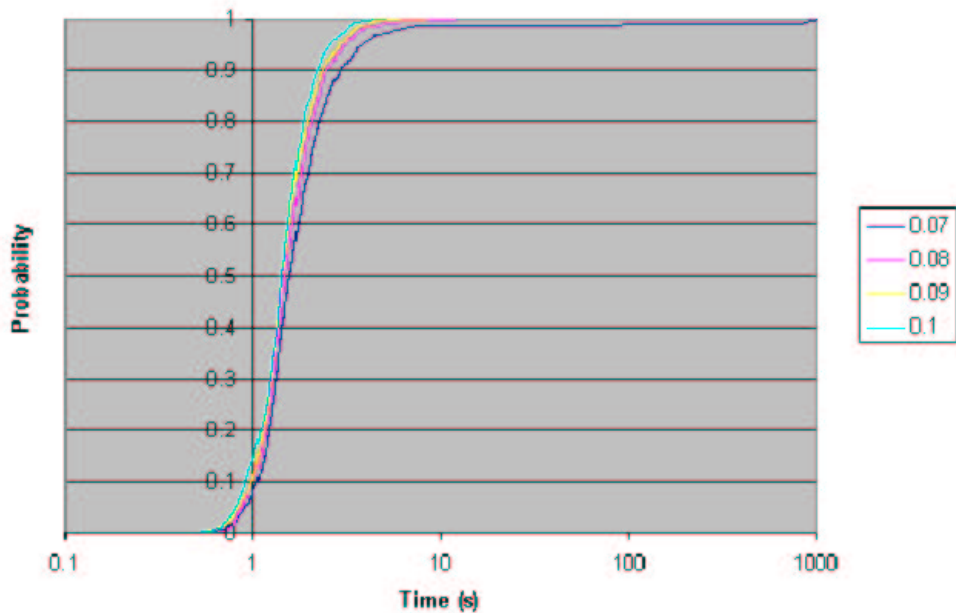
74

Figure 14: A comparison of the effects of RTOL on the running time of DS

for stopping before averaging, such as Euclidean distance traveled.

### 3.1.6    DS

In our tests DS was unable to converge to a result that was visually indistinguishable from the original BRDF. An examination of a typical run of DS shows that when all of the points are contracted in a single move, the highest error in the current simplex usually becomes significantly worse. As a consequence of this, the threshold for the ratio between the highest and the lowest objective function values used for termination is extremely important. If this threshold is set too low then it takes a long time for the method to stop without any noticeable improvement in the solution quality. The time required for DS to stop based on the ratio is very sensitive to the cutoff point, RTOL. For example, a single run of DS with RTOL set to 0.02 for the synthetic two specular lobes instance with no noise was not able to converge after 1000 minutes of CPU time. However, runs conducted with RTOL set to 0.03 typically finished in under a minute.

In 1000 runs on the single lobe synthetic BRDF with Gaussian noise of 0.10, DS required over 700 seconds for 10 runs when RTOL=0.07. All 1000 runs were finished in less than 12 seconds each when RTOL=0.08. See Figure 14 for the effect of RTOL on the runtime.

However, changing RTOL in the range between 0.07 and 0.10 had very little effect on the solution quality as evidenced in Table 5.

Although the solution quality increases slightly as RTOL is decreased, dropping RTOL below the empirically defined threshold for this problem does not lead to significant gains. This threshold is

75

|        | RTOL = 0.07 | RTOL = 0.08 | RTOL = 0.09 | RTOL = 0.10 |
|--------|-------------|-------------|-------------|-------------|
| Mean   | 0.149285    | 0.150191    | 0.152539    | 0.153406    |
| Median | 0.132279    | 0.132467    | 0.132935    | 0.133564    |
| Min    | 0.129501    | 0.129754    | 0.129494    | 0.129603    |
| Max    | 0.248545    | 0.248888    | 0.352014    | 0.256441    |

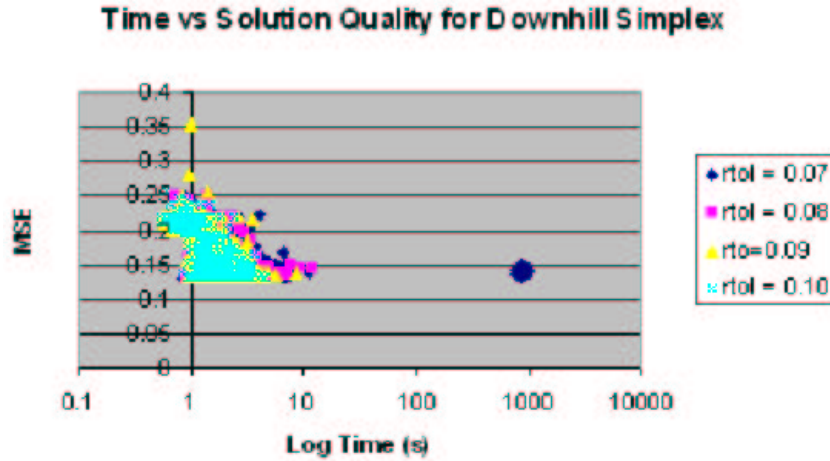Table 5: Effect of RTOL on RMSE with single specular lobe synthetic data used for tuning.



Figure 15: A comparison between running time and solution quality for DS. Note the points that are generated as a result of RTOL dropping below the threshold.

related to the roughness of the search space. As more noise was added to the problem, the threshold value increased. Figure 15 shows the relationship between the time spent on the problem and the solution quality. When RTOL is above the threshold, the time values are very low with a significant spread on the solution quality. When RTOL drops below the threshold the solution quality stays roughly constant and the time required to reach a solution varies.

As RTOL approaches the threshold the solutions that are found take longer to find but are consistently better. This intuitively makes sense because as RTOL approaches infinity the method becomes closer and closer to random picking.

Modifying DS to accept worsening steps and to select a random vertex with some probability $p$ are possible improvements to this algorithm.

## 3.2   Results With Synthetic Data

Figure 16 is the result of our experiments with synthetic data. Each RTD is generated with more than 500 independent runs, except ASA on the crossing specular lobe cases because it was only feasible to do 100 runs; approximately half of these runs exceeded the 20 CPU minute time cut-

off. On all synthetic instances AED and AERS probabilistically dominate the naive RPLS and sophisticated ASA. On the untuned cases there is a clear ordering of AED, AERS, RPLS, and ASA from probabilistically fastest to slowest.
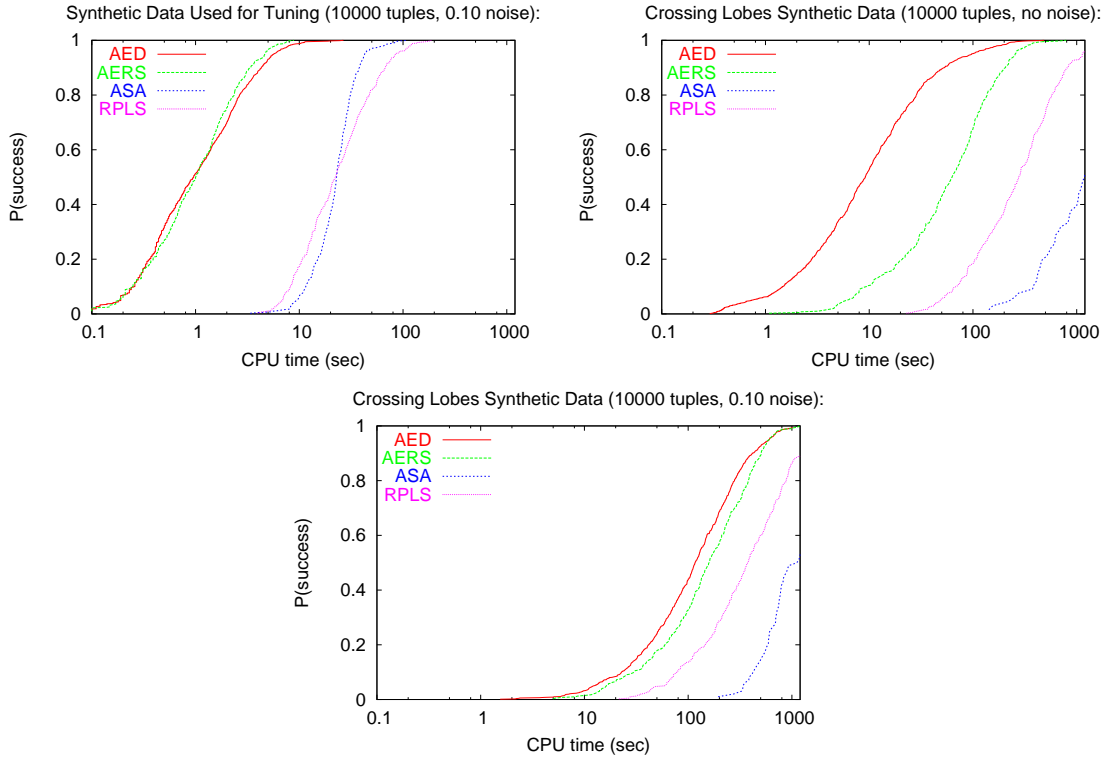


Figure 16: Results on sample data generated by a Lafortune model BRDF.

## 3.3 Real Data

Our single real world test case is the isotropic blue latex house paint BRDF measured at 550 nm and approximated in [4]. We used the interpolated data set available at `http://www.graphics.cornell.edu/online/measurements/reflectance-/housepaints/index.html` rather than the raw data set. However, even this interpolated data set has large outliers and we had to remove BRDF values exceeding 1 to get a reasonable fit. With this filtering our threshold RMSE of 0.042 was substantially lower than the RMSE of 0.147 that the Lafortune model approximation in [4] achieved. Although the model in [4] has 3 specular lobes, we found that using more than 1 specular lobe did not improve the RMSE quality of the fit.

Unfortunately, none of the 200 runs of ASA reached the threshold RMSE of 0.042 within 20 CPU minutes and we were not able to include ASA in Figure 17. Closer evaluation of algorithm traces indicates that ASA typically stagnates with a RMSE greater than 0.05.

On this data set AERS well outperforms AED because AED repeatedly increases its subsample

77

size until it restarts, which accounts for the shape of the RTD in Figure 17.
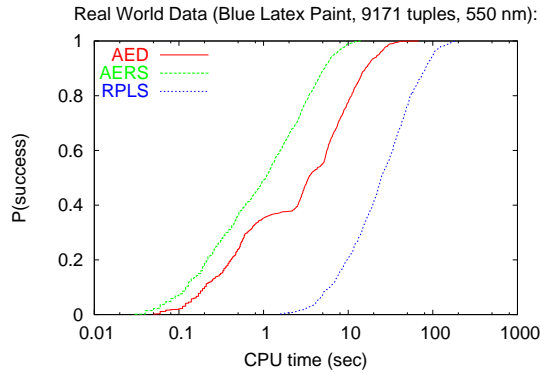


Figure 17: Blue house paint studied in [4].

The polar plots in Figure 18 represent the measured BRDF data, the Lafortune model approximation presented in [4], and the BRDF in Table 6 which was found by AED and is representative of BRDFs with a single specular lobe with RMSE below the threshold of 0.042. In these plots the surface normal, origin, incident vector, and exitant vector all lie in the same plane like the 2-dimensional Figure 1. Each plot has a different incident vector given by its angle with the surface normal. The magnitude of the polar coordinate system is the BRDF and the angle of the coordinate system is the angle between the normal and the exitant vector. Although our fitted BRDF has a lower RMSE than the Lafortune's approximation, the his approximation is clearly a better fit in the last 2 slices because it more closely follows the true curve. However, $R_m$ neither Lafortune's approximation nor our BRDF adequately captured the forward reflection at the grazing angle of $65°$.

| Lobe | $C_x = C_y$ | $C_z$ | pan | n |
|---|---|---|---|---|
| I | 0.914 | 0.52 | 0° | 10.21 |
| Diffuse | 0.126 | | | |

Table 6: Lafortune BRDF parameters the fitted BRDF in Figure 18.

## 4    Conclusions and Further Research

Random subsampling is an effective optimization on all of our test cases. Intuitively, this is because repeated subsampling smooths the search landscape by washing away local minima caused by individual data samples. However, it is unclear if adaptively changing the subsample size is useful. The adaptive parameters of AED were no more robust than the fixed parameter in AERS. On the real world data where AERS outperforms AED an interesting phenomenon occurs: AED repeatedly exceeds its maximum subsample size and restarts. This suggests that introducing an asymmetry in the rate of subsample increase and decrease may be worthwhile. Moreover, a less
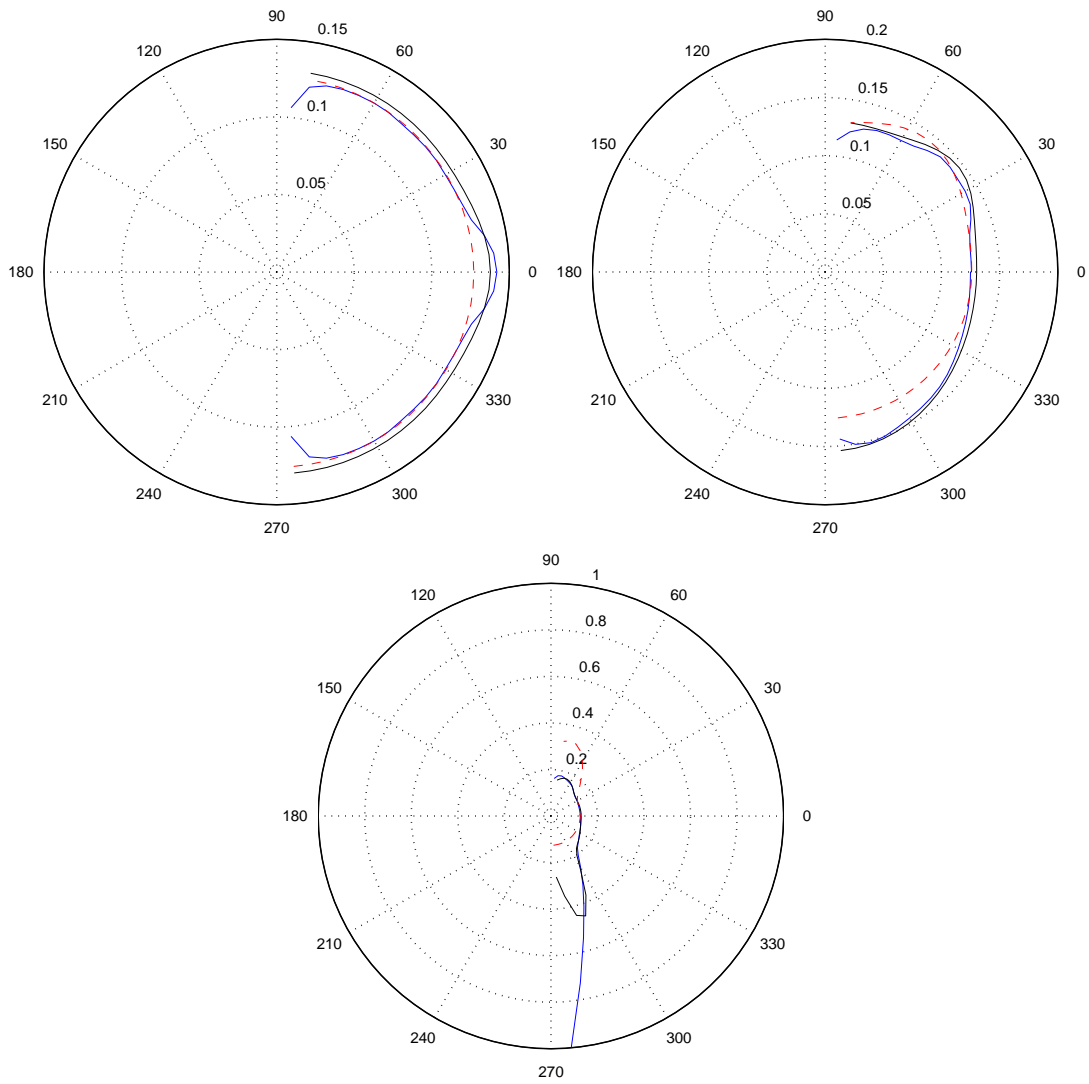
Figure 18: Measured BRDF (blue), Lafortune approximation (black), and our fitted BRDF (red) for blue latex paint at incident angles of $0°$, $35°$, and $65°$ from the normal.

reactive subsample size adjustment mechanism such as an arithmetic progression instead of geometric progression is worth investigating.

The issue of measuring the quality of fit is a difficult problem because many interesting effects occur at grazing angles; angles which in turn cause large measurement errors. More samples will average out measurement errors at the expense of computational complexity and roughening the search landscape. As we witnessed, the qualitative objective of achieving a good visual fit is only approximated with the quantitative RMSE.

Simulated annealing caused endless headaches and performed poorly. As tuning is more art than science, our inexperience may have caused many of our problems, but this was only exacerbated by the plethora of algorithm parameters. Although ASA took much longer to converge compared to AED and AERS, its performance was still better than that of the SA-based algorithms, SALS and AVG. This is likely due to the former's more complex perturbation mechanism, based on local gradients of the objective function, and to its non-uniform adaptive cooling schedule. The performance of SA can perhaps be further improved by taking into account both the importance and wide range of the specularity parameter. Possible ideas include: using separate temperature scaling for the specularity values; discretizing the search space and performing perturbations on smaller subranges; maintaining a history of previous specularity values and using it to guide the perturbations in specific directions.

# References

[1] Jonathan Backer. BRDF Fitting. CPSC 514 Course Project Fall 2002/03. `http://www.cs.ubc.ca/~backer/index.html`

[2] Holger H. Hoos and Thomas Stützle. Stochastic Local Search - Foundations and Applications. Morgan Kaufmann Publishers, to appear.

[3] L. Ingber. Adaptive simulated annealing (ASA): Lessons learned. Control and Cybernetics, volume 25, pages 33-54, 1996. `http://www.ingber.com/asa96_lessons.pdf`

[4] Eric P. F. Lafortune, Sing-Choong Foo, Kenneth E. Torrance, and Donald P. Greenberg. Non-linear approximation of reflectance functions. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 117–126. ACM Press/Addison-Wesley Publishing Co., 1997.

[5] Timothy Masters. Advanced Algorithms for Neural Networks. John Wiley & Sons, 1995.

[6] Bui Tuong Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975.

[7] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling Numerical Recipes: The Art of Scientific Computing, 2nd Edition Cambridge University Press, 1992.

[8] Christophe Schlick. A survey of shading and reflectance models. *Computer Graphics Forum*, 13(2):121–131, 1994.