Emumerating small hyperbolic 3-manifolds

by

Nathaniel Thurston

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Mathematics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Robion Kirby, Chair
Professor David Gabai
Associate Professor Henri Poincaré

Summer 2022

Emumerating small hyperbolic 3-manifolds

Abstract

This paper documents a computer-assisted procedure for rigorously analyzing small hyperbolic 3-manifolds. Briefly, we will define a compact six-dimensional space $\mathcal{P}$ that parameterizes pairs of elements of $\mathrm{I}som(\mathbf{H}^3)$, and then construct a regular binary space partition (BSP) tree which subdivides $\mathcal{P}$ into subregions $\mathcal{P}_i$ and whose leaves are — with a few exceptions — labeled with killerwords. These killerwords will encode miniature proofs that $\mathcal{P}_i$ cannot contain any points which correspond to particular choices of pairs of generators of any torsion-free discrete group of $\mathrm{I}som(\mathbf{H}^3)$. This tree of mini-proofs will then be used to exhaustively isolate all possible manifolds which can have properties related to the dimensions of $\mathcal{P}$.

This method has been used as a foundation for tackling multiple long-standing problems at the interface of hyperbolic geometry and low-dimensional topology. In particular, topological rigidity of hyperbolic 3-manifolds, the generalized Smale conjecture for hyperbolic 3-manifolds, finding the closed 3-manifold of least volume, and the Gordon conjecture on exceptional Dehn fillings.

Chapter 1

Introduction

This paper documents a computer-assisted procedure for rigorously analyzing small hyperbolic 3-manifolds. Briefly, we will define a compact six-dimensional space $\mathcal{P}$ that parameterizes pairs of elements of I$som(\mathbf{H}^3)$, and then construct a regular binary space partition (BSP) tree which subdivides $\mathcal{P}$ into subregions $\mathcal{P}_i$ and whose leaves are — with a few exceptions — labeled with killerwords. These killerwords will encode miniature proofs that $\mathcal{P}_i$ cannot contain any points which correspond to particular choices of pairs of generators of any torsion-free discrete group of I$som(\mathbf{H}^3)$. This tree of mini-proofs will then be used to exhaustively isolate all possible manifolds which can have properties related to the dimensions of $\mathcal{P}$.

The first two applications of the procedure were used to prove a proposition from [GMT]:

Proposition 1.1. *[GMT] Let M be an orientable hyperbolic 3-manifold, and let δ be a shortest geodesic. Then, either tuberadius$(\delta) > \ln(3)/2$, or M lies within one of seven tightly constrained exceptional shortest-geodesic-geometry regions.*

This proposition and a related one were used in the proof of the main technical result of [GMT], the topological rigidity of hyperbolic 3-manifolds.

Theorem 1.2. *[GMT] Every closed hyperbolic 3-manifold has a non-coalescible insulator family, indeed one coming from a shortest geodesic. As a consequence, homotopy hyperbolic 3-manifolds are hyperbolic.*

More recently, the proposition has been sharpened:

Theorem 1.3. *Let M be an orientable hyperbolic 3-manifold, and let δ be a shortest geodesic. Then, either tuberadius$(\delta) > \ln(3)/2$, or M is isometric to one of seven specific manifolds.*

Proof. Let $X_0, \cdots, X_6$ denote the exceptional regions of Proposition 2.1 with N_i the corresponding conjectural manifold. [GMT] showed that N_0=Vol3 is the unique manifold in the region and [JR] showed that Vol3 covered no 3-manifold. [JR] also proved that N_5 and N_6 are isometric. [CLLMR] and [L] identify a manifold in each region and [CLLMR] show that

these manifolds are the unique ones in its region. [CLLMR] show that N_1, N_5 and N_6 cover no manifold. In [GT] the proof is completed by showing that N_3 covers no manifold and each of N_2 and N_4 2-fold cover manifolds, however the quotients are all non exceptional, i.e. each shortest geodesic has a $ln(3)/2$ tube. □

Other applications of the procedure:

Theorem 1.4. *[G1] (Smale conjecture of hyperbolic 3-manifolds) If N is a closed hyperbolic 3-manifold, then the natural inclusion* $\mathrm{Isom}(N) \to \mathrm{Diff}(N)$ *is a homotopy equivalence.*

The proof makes essential use of the fact that a shortest geodesic of a closed hyperbolic 3-manifold satisfies the insulator condition [GMT].

Theorem 1.5. *[GMM], [Mi] The Weeks manifold is the unique closed orientable hyperbolic 3-manifold of minimal volume.*

In addition to [GMT] this result relies on

Lemma 1.6. *[ACS] Suppose that M is a closed, orientable hyperbolic 3-manifold, and that C is a shortest geodesic in M. Set* $N = \mathrm{drill}_C(M)$. *If* $\mathrm{tuberad}(C) \geq \ln(3)/2$ *then* $\mathrm{vol}N < 3.0177\mathrm{vol}M$.

This is based on a result of [ADST], that makes essential use of Perelman's [Pe1], [Pe2] Ricci flow with surgery and as well as his monotonicity result, which are key elements of his proof of geometrization. Lemma 1.6 is used to prove other results such as:

Theorem 1.7. *[ACS] Suppose that M is a closed, orientable, hyperbolic 3-manifold such that* $\mathrm{vol}(M) \leq 1.22$. *Then* $H_1(M : Zp)$ *has dimension at most 3 for every prime p.*

That result also requires this work and [GMT].

The methods of this work are crucial to the following result.

Theorem 1.8. *[GHMTY] The figure-8 knot complement is the unique 1-cusped hyperbolic 3-manifold with 9 or more non hyperbolic fillings.*

This result gives a positive proof of the long-standing Gordon exceptional filling conjecture that has attracted much attention. See [GHMTY] for a detailed history. An important result in this direction was the theorem of Lackenby and Meyerhoff [LM] who showed that a cusped hyperbolic 3-manifold has at most 10 exceptional fillings. That result also relied on rigorous computer assistance using the AffApprox technology.

For a survey this work and further developments including other applications, see [GMTY].

This paper is organized as follows.
In Chapter 2 we introduce [GMT]; Proposition 1.28 as Theorem 2.1, and sketch its proof.

In Chapter 3 we provide a formal statement and proof of Theorem 2.1.

In Chapter 4, the method for describing the decomposition of the parameter space $\mathcal{W}$ into sub-regions is given, and the conditions used to eliminate sub-regions are discussed. Near the end of this section, the first part of a detailed example is given.

Eliminating a sub-region requires that a certain function is shown to be bounded appropriately over the entire sub-region. This is carried out by using a first-order Taylor approximation of the function together with a remainder bound. Our computer version of such a Taylor approximation with remainder bound is called an *A*ffApprox and in Chapter 5, the relevant theory is developed. At this point, the detailed example of Chapter 4 can be completed.

In Chapter 6 and 7, round-off error analysis appropriate to our set-up is introduced. Specifically, in Chapter 7, round-off error is incorporated into the *A*ffApprox formulas introduced in Chapter 5. The proofs here require an analysis of round-off error for complex numbers, which is carried out in Chapter 6.

In Section 8 we give some hints about the search for a tree, in the hope that they will help others endeavoring to apply these methods.

Finally, in Chapter 9, we present an updated version of the code used to check that the proof is valid, with self-contained copies of the proofs required for the reader to check its own validity.

Prior publication: Chapter 2 includes material from section 0 of [GMT]. Chapters 3 through 7 originally appeared as sections 1 and 5 through 8 of [GMT], and are reproduced here with minor revision. In particular, Chapter 3 is a somewhat abridged version of section 1 of [GMT]. Chapter 8, which includes new insights, also contains material which originally appeared in remarks in [GMT].

Acknowledgements:

Chapter 2

Technical Introduction

Here is a brief description of why Proposition 1.2 might be amenable to computer-assisted proof. If a shortest geodesic δ in a hyperbolic 3-manifold N does not have a $\ln(3)/2$ tube then there is a 2-generator subgroup G of $\pi_1(N) = \Gamma$ which also does not have that property. Specifically, take G generated by f and w, with $f \in \Gamma$ a primitive hyperbolic isometry whose fixed axis $\delta_0 \subset \mathbf{H}^3$ projects to δ, and with $w \in \Gamma$ a hyperbolic isometry which takes δ_0 to a nearest translate. Then, after identifying $N = \mathbf{H}^3/\Gamma$ and letting $Z = \mathbf{H}^3/G$, we see that the shortest geodesic in Z (which corresponds to δ) does not have a $\ln(3)/2$ tube. Thus, to understand solid tubes around shortest geodesics in hyperbolic 3-manifolds, we need to understand appropriate 2-generator groups, and this can be done by a parameter space analysis as follows. (Parameter space analyses are naturally amenable to computer proofs.)

The space of *r*elevant (see Definition 3.10) 2-generator groups in $\mathrm{I}som_+(\mathbf{H}^3)$ is naturally parametrized by a subset $\mathcal{P}$ of $\mathbf{C}^3$. Each parameter corresponds to a 2-generator group G with specified generators f and w, and we call such a group a *m*arked group. The marked groups of particular interest are those in which G is discrete, torsion-free, parabolic-free, f corresponds to a shortest geodesic δ, and w corresponds to a covering translation of a particular lift of δ to a nearest translate. We denote this set of particularly interesting marked groups by $\mathcal{T}$. We show that if $\mathrm{tuberadius}(\delta) \leq \ln(3)/2$ in a hyperbolic 3-manifold N, then G must correspond to a parameter lying in one of seven small regions $\mathcal{R}_n,\ n = 0, \ldots, 6$ in $\mathcal{P}$. With respect to this notation, we have:

Proposition 2.1. $\mathcal{T} \cap (\mathcal{P} - \cup_{n=0,\ldots,6}\mathcal{R}_n) = \emptyset$.

The full statement of Proposition 2.1 explicitly describes the seven small regions of the parameter space as well as some associated data.

Here is the idea of the proof. Roughly speaking, we subdivide $\mathcal{P}$ into a billion regions of varying sizes, and show that all but the seven exceptional regions cannot contain a parameter corresponding to a "shortest/nearest" marked group. For example we would know that a region $\mathcal{R}$ contained no such group if we knew that for each point $\rho \in \mathcal{R}$, $\text{Relength}(f_\rho) > \text{R}elength(w_\rho)$. (Here $\text{Relength}(f_\rho)$ (resp. $\text{Relength}(w_\rho)$) denotes the real translation length of the isometry of $\mathbf{H}^3$ corresponding to the element f (resp. w) in the marked group with parameter ρ.) This inequality would contradict the fact that f corresponds to δ which is a *s*hortest geodesic. Similarly, there are *n*earest contradictions.

Chapter 3

Killerwords and the parameter space

Notations And Conventions 3.1. . A hyperbolic 3-manifold is a Riemannian 3-manifold of constant sectional curvature -1. All hyperbolic 3-manifolds under consideration will be closed and orientable. We will work in the upper-half-space model for hyperbolic 3-space: $\mathbf{H}^3 = \{(x, y, z) : z > 0\}$ with metric $\mathrm{d}s_H = \mathrm{d}s_E/z$. The distance between two points w and v in $\mathbf{H}^3$ will be denoted $\rho(w, v)$.

It is well known that $\mathrm{I}som_+(\mathbf{H}^3) = \mathrm{P}SL(2,\mathbf{C})$, where an element of $\mathrm{P}SL(2,\mathbf{C})$ acts as a Möbius transformation on the bounding (extended) complex plane and the extension to upper-half-space is the natural extension (see [Bea]). If M is a hyperbolic 3-manifold, then $M = \mathbf{H}^3/\Gamma$ where Γ is a discrete, torsion-free subgroup of $\mathrm{P}SL(2,\mathbf{C})$.

For computational convenience, we will often normalize so that the (positive) z-axis is the axis of an isometry. As such, we set up some special notation. Let $B_{(0;\infty)}$ denote the oriented geodesic $\{(0, 0, z) : 0 < z < \infty\}$, with negative endpoint $(0, 0, 0)$. (An *e*ndpoint of an axis refers to a limit point of the axis on S^2_∞.) Let $B_{(-1;1)}$ denote the oriented geodesic with negative endpoint $(-1, 0, 0)$ and positive endpoint $(1, 0, 0)$.

When working in a group G generated by f and w and looking at words in f, w, f^{-1}, w^{-1} we will often let F and W denote f^{-1} and w^{-1}, respectively.

Definition 3.2. If f is an isometry, then we define

$$\mathit{Relength}(f) = \inf\{\rho(w, f(w)) \mid w \in \mathbf{H}^3\}.$$

Thus $\mathrm{R}\mathit{elength}(f) = 0$ if and only if f is either a parabolic or elliptic isometry. If $\mathrm{R}\mathit{elength}(f) > 0$, then f is hyperbolic and maps a unique geodesic σ in $\mathbf{H}^3$ to itself. In that case σ is oriented (the negative end being the repelling fixed point on S^2_∞) and the isometry f is the composition of a rotation of t (m*od* 2π) radians along σ (the sign of the angle of rotation is determined by the right-hand rule) followed by a pure translation of $\mathbf{H}^3$ along σ of $l = \mathrm{R}\mathit{elength}(f)$. We define $\mathit{length}(f) = l + it$, and call $A_f = \sigma$ the *a*xis of f. Now, A_f is an oriented interval with endpoints in S^2_∞, the orientation being induced from σ.

If the geodesic σ is given a fixed orientation, we define an $l + it$ translation f along σ to be a distance l translation in the positive direction, followed by a rotation of σ by t radians.

Of course if $l < 0$, then each point of σ gets moved $-l$ in the negative direction. Also, via the right-hand rule, the orientation determines what is meant by a t-radian rotation. Thus if $l > 0$, the orientation induced on σ by f (as in the previous paragraph) equals the given orientation. If $l < 0$, then the induced orientation is opposite to the given orientation and f is a $-(l + it)$ translation of $-\sigma$ in the sense of the previous paragraph.

If f is elliptic, then f is a rotation of t radians where $0 \leq t \leq \pi$ about some oriented geodesic, and we define $length(f) = ti$. If f is parabolic or the identity, we define $length(f) = 0 + i0$. So, for all isometries we have that $Relength = Re(length)$.

Definition 3.3. If G is a subgroup of $Isom_+(\mathbf{H}^3)$, then we say that f is a *s*hortest element in G if $f \neq id$ and $Relength(f) \leq Relength(g)$ for all $g \in G,\ g \neq id$.

Definition 3.4. If $\sigma,\ \tau$ are disjoint oriented geodesics in $\mathbf{H}^3$ which do not meet at infinity, then define $distance(\sigma, \tau) = length(w)$ where $w \in Isom_+(\mathbf{H}^3)$ is the hyperbolic element which translates $\mathbf{H}^3$ along the unique common perpendicular between σ and τ and which takes the oriented geodesic σ to the oriented geodesic τ. The oriented common perpendicular from σ to τ is called the *o*rthocurve between σ and τ. The *o*rtholine between σ and τ is the complete oriented geodesic in $\mathbf{H}^3$ which contains the orthocurve between σ and τ.

If σ and τ intersect at one point in $\mathbf{H}^3$ then there is an elliptic isometry w taking σ to τ fixing $\sigma \cap \tau$. Again, define $distance(\sigma, \tau) = length(w)$. In this case, the orthocurve is the point $\sigma \cap \tau$, and the ortholine O from σ to τ is oriented so that $\sigma,\ \tau,\ O$ form a right-handed frame.

If σ and τ intersect at infinity, then there is no unique common perpendicular, hence no ortholine, and we define $distance(\sigma, \tau) = 0 + i0$, or $0 + i\pi$ depending on whether or not σ and τ point in the same direction at their intersection point(s) at infinity.

Define $Redistance = Re(distance)$.

As defined, Redistance is nonnegative. In Definition 3.6, it will be useful to have a broader definition. Given an oriented geodesic α in $\mathbf{H}^3$ orthogonal to oriented geodesics β and γ, define $d_\alpha(\beta, \gamma) \in \mathbf{C}$ where a $d_\alpha(\beta, \gamma)$ translation of $\mathbf{H}^3$ along α takes β to γ.

Definition 3.5. A tube of radius r about a geodesic δ_0 in $\mathbf{H}^3$ is $\{w \in \mathbf{H}^3 \mid \rho(w, v) \leq r$ for some $v \in \delta_0\}$. If δ is a simple closed geodesic in the hyperbolic 3-manifold N and if $\{\delta_i\}$ is the set of pre-images of δ in $\mathbf{H}^3$, then define $tuberadius(\delta) = \frac{1}{2} \min\{Redistance(\delta_i, \delta_j) \mid i \neq j\}$. If $r = tuberadius(\delta)$, then define a *m*aximal tube about δ to be the image of a tube of radius r about δ_0. Note that tuberadius$(\delta) = \sup\{r \mid$ there exists an embedded tubular neighborhood of radius r about $\delta\}$.

Definition 3.6. Our desire to understand tuberadii about closed geodesics, and especially about a simple closed geodesic δ, leads us to investigate certain 2-generator subgroups $G = \langle f, w\rangle$ of $Isom_+(\mathbf{H}^3)$ with the generator f corresponding to a primitive isometry fixing δ_0 and the generator w corresponding to an element taking δ_0 to its nearest covering translate. We investigate these 2-generator groups by using certain subsets of $\mathbf{C}^3$ as parameter spaces.

A *m*arked (2-generator) group is a triple $\{G, f, w\}$ consisting of a 2-generator subgroup G of $Isom_+(\mathbf{H}^3)$ and an ordered pair of isometries f, w of $\mathbf{H}^3$ which generate G such that $Relength(f) > 0$ and if A_f is the axis of f, then $w(A_f) \cap A_f = \emptyset$ (here, intersection is taken in $\mathbf{H}^3 \cup S^2_\infty$). Two marked groups $\{G_1, f_1, w_1\}$ and $\{G_2, f_2, w_2\}$ are *c*onjugate if G_1 and G_2 are conjugate via an element of $Isom_+(\mathbf{H}^3)$ and this conjugating element takes f_1 to f_2 and w_1 to w_2. Within any conjugacy class of marked groups is a unique normalized element $\{G, f, w\}$ where f is a positive translation along the (oriented) geodesic $B_{(0;\infty)}$, and the orthocurve from $w^{-1}(B_{(0;\infty)})$ to $B_{(0;\infty)}$ lies on $B_{(-1;1)}$ on the negative side of $B_{(-1;1)} \cap B_{(0;\infty)}$. To minimize notation, we will frequently equate a conjugacy class with its normal representative.

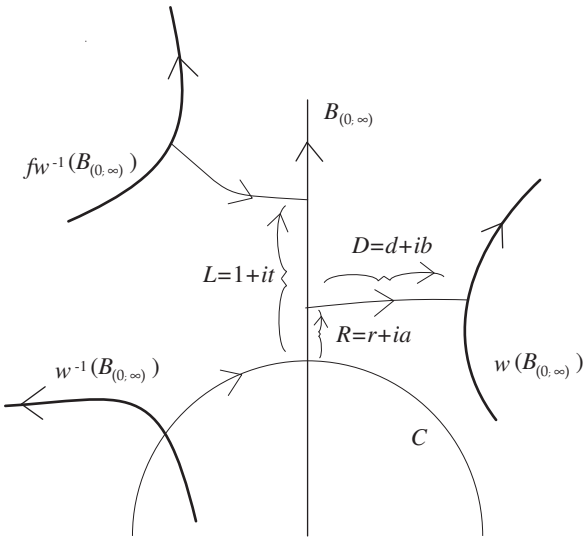
Given $(L, D, R) = (l + it, d + ib, r + ia) \in \mathbf{C}^3$ with $l > 0,\ d > 0$, one can associate a group G generated by elements f and w as follows. Define f to be an $l + it$ translation along $B_{(0;\infty)}$ and w to be a $d + ib$ translation along $B_{(-1;1)}$ followed by an $r + ia$ translation along $B_{(0;\infty)}$ (here, r can be negative, in which case this is equivalent to a $-r - ia$ translation along $-B_{(0;\infty)}$). Conversely if $\{G, f, w\}$ is a normalized marked group then f is an L translation of $B_{(0;\infty)}$ and w is a D translation of $B_{(-1;1)}$ followed by an R translation of $B_{(0;\infty)}$. Thus $\mathcal{P}' = \{(l + it, d + ib, r + ia) \in \mathbf{C}^3 |\ l > 0, d > 0\}$ parametrizes the set of conjugacy classes of marked groups. In particular, the parametrization is surjective and locally one-to-one.

We are primarily interested in the set $\mathcal{T}' \subset \mathcal{P}'$ which parametrizes all conjugacy classes of marked groups $\{G, f, w\}$ for which f is a shortest element of G which (positively) translates $B_{(0;\infty)}$ and $w \in G$ takes $B_{(0;\infty)}$ to a *n*earest translate $w(B_{(0;\infty)})$ such that $-Relength(f)/2 < Re(d_{B_{(0;\infty)}})$ (ortholine from $w^{-1}(B_{(0;\infty)})$ to $B_{(0;\infty)}$), (ortholine from $B_{(0;\infty)}$ to $w(B_{(0;\infty)})$)))) $\leq Relength(f)/2$. See Figure 3. Note that because f is shortest and $Relength(f) > 0$, it follows that G must be discrete, torsion-free, and parabolic-free.

Remark 3.7. $\mathcal{T}'$ consists of those parameters corresponding to marked groups $\{G, f, w\}$ such that l is the real length of a shortest element of G, d is the real distance between $B_{(0;\infty)}$ and a nearest translate, and $-l/2 < r \leq l/2$. In what follows, it is essential to remember that an element α of $\mathcal{P}'$ corresponds not only to a group G, but to a marked group. To further establish the point, we note that, for elements of $\mathcal{T}'$, the parameter l is an invariant of G alone (that is, l is the shortest real length of an element of G), while the parameter d is determined by G and f (that is, the notion of "nearest" used to define w in the definition of $\mathcal{T}'$ requires a choice of f).

As mentioned in the introduction to this paper, we are only interested in the subset of $\mathcal{T}'$ corresponding to parameters α with $d \leq \ln(3)$. The following two propositions imply this subset of $\mathcal{T}'$ lives in a compact subset of $\mathcal{P}'$.

Proposition 3.8. *All closed geodesics of length less than* 0.0979 *in all hyperbolic* 3*-manifolds have (embedded) tubes of radius* $\ln(3)/2$.



Proof. In [M1] it is proved that a closed geodesic of length $x+iy$ has a tube (embedded) of radius $r(x+iy)$ satisfying

$$\sinh^2(r(x+iy)) = \max_{n\in\mathbf{Z}_+} \frac{1}{2}\left(\frac{\sqrt{1-2k(x,y,n)}}{k(x,y,n)} - 1\right)$$

where

$$k(x,y,n) = \cosh(nx) - \cos(ny),$$

where we restrict to $x+iy$ values which produce positive radii $r(x+iy)$ by means of this formula. It is easy to compute that for a given $x+iy$ we need to have n for which $0 < k(x,y,n) < \sqrt{2}-1$ to produce a positive radius tube by this method.

The function $\frac{1}{2}(\frac{\sqrt{1-2k}}{k}-1)$ is decreasing on the interval $(0,-1+\sqrt{2})$. It is easy to solve for the k value that produces radius $r = \ln(3)/2$ and it is just over 0.3397. Thus, by restricting to k values in the interval $(0, 0.3397)$ we guarantee radii r greater than $\ln(3)/2$.

Thus, to complete the proof of this proposition, we need to show that when a closed geodesic has real length x less than 0.0979, there exists a positive integer n for which $k(x,y,n)$ is less than 0.3397 for all angles y. Because cosh is an increasing function, we can restrict our analysis to $x = 0.0979$. Thus, we need only show that given any angle y, we can find a

positive integer n such that $\cosh(n0.0979) - \cos(ny) < 0.3397$. When $n > 8$ we can compute that $\cosh(n0.0979) - \cos(ny) > 0.3397$, and we therefore restrict to positive integers $n \leq 8$.

We now consider angles y. Because cos is an even function, we need only consider $y \in [0, \pi]$. Finally, we complete the proof by covering $[0, \pi]$ by 11 overlapping closed sub-intervals σ_i each of which has an associated positive integer n_i for which $\cosh(n_i 0.0979) - \cos(n_i y) < 0.3397$ is true for all $y \in \sigma_i$:

$$\begin{array}{llll} \sigma_0 = [0.000, 0.843] & n_0 = 1 & \sigma_5 = [1.733, 1.858] & n_5 = 7 \\ \sigma_1 = [0.835, 0.960] & n_1 = 7 & \sigma_6 = [1.832, 2.357] & n_6 = 3 \\ \sigma_2 = [0.951, 1.143] & n_2 = 6 & \sigma_7 = [2.334, 2.3792] & n_7 = 8 \\ \sigma_3 = [1.123, 1.391] & n_3 = 5 & \sigma_8 = [2.3789, 2.647] & n_8 = 5 \\ \sigma_4 = [1.386, 1.755] & n_4 = 4 & \sigma_9 = [2.630, 2.755] & n_9 = 7 \end{array}$$

$$\sigma_{10} = [2.730, \pi] \quad n_{10} = 2$$

□

Proposition 3.9. *A shortest geodesic δ in a closed hyperbolic 3-manifold has*

i) $tuberadius(\delta) \geq l/4$ where $l = Relength(\delta)$, and

ii) $tuberadius(\delta) > \ln(3)/2$ if $Relength(\delta) \geq 1.289785$.

Proof. Part i) is a consequence of the following well-known argument: Uniformly expand a tube around a shortest geodesic in the hyperbolic 3-manifold. If the expanding tube hits itself before a radius of $l/4$ then we will construct a loop of length less than l, which produces a contradiction to being shortest. Drop the two obvious perpendiculars from the hitting point down to the core geodesic. Consider the following loop—down one perpendicular, follow the shorter direction on the core geodesic, up the other perpendicular. Because a lift of this loop to $\mathbf{H}^3$ is not closed, this loop is homotopically nontrivial. By construction it has length less than $l/4 + l/2 + l/4 = l$.

To prove part ii), we improve on this loop. Replace the first half of the journey by the hypotenuse of the right triangle formed by the first perpendicular and the first half of the shorter arc along the core geodesic. Replace the second half of the journey by the hypotenuse of the right triangle formed by the second perpendicular and the second half of the shorter arc along the core geodesic. Now, to analyze the specific case of tuberadius $\ln(3)/2$, we use the hyperbolic Pythagorean theorem (see [F]) $\cosh c = (\cosh a)(\cosh b)$ with $a = \ln(3)/2$ and $b = l/4$. We get that the length of the constructed loop is $2\mathrm{A}rccosh(\cosh(\ln(3)/2)\cosh(l/4))$ and this is less than l when $l > 1.289784\ldots$, by a calculation (which follows) and the fact that

$$2\mathrm{A}rccosh(\cosh(\ln(3)/2)(\cosh(l/4)) - l$$

is a decreasing function of l.

We solve explicitly for the value of l at which

$$2\mathrm{A}rccosh(\cosh(\ln(3)/2)(\cosh(l/4)) - l = 0.$$

Noting that $\cosh(\ln(3)/2) = \frac{2}{\sqrt{3}}$ we get $\frac{2}{\sqrt{3}}(\cosh(l/4)) = \cosh(l/2)$. Using a half-angle formula for $\cosh(l/2)$ we get $\frac{2}{\sqrt{3}}(\cosh(l/4)) = 2\cosh^2(l/4) - 1$. Setting $x = \cosh(l/4)$ we get the quadratic $\frac{2}{\sqrt{3}}x = 2x^2 - 1$. Solving and substituting, we get $l = 1.289784\ldots$ □

Definition 3.10. Let $\mathcal{P} \subset \mathcal{P}'$ be the set of those parameters $\alpha = (l + it,\, d + ib, r + ia)$ such that

a) $0.0978 \leq l \leq 1.289785$,
b) $l/2 \leq d \leq \ln(3)$,
c) $0 \leq r \leq l/2$,
d) $-\pi \leq t \leq 0$,
e) $-\pi \leq b \leq \pi$,
f) $-\pi \leq a \leq \pi$.

Define $\mathcal{T} = \mathcal{T}' \cap \mathcal{P}$.

The point of the definition of $\mathcal{P}$ and $\mathcal{T}$ is as follows. We want to analyze by computer the relationship between lengths of shortest geodesics and their tuberadii in hyperbolic 3-manifolds. We were naturally led to the parameter space $\mathcal{P}'$ and its subset $\mathcal{T}'$. But $\mathcal{P}'$ is problematic from the computational viewpoint because it is noncompact. We wish to replace $\mathcal{P}'$ by $\mathcal{P}$ which is compact, and $\mathcal{T}'$ by $\mathcal{T}$ in our computer analysis. This is carried out in Lemma 3.11. Note that we worked to make $\mathcal{P}$ as small as reasonable to save computation time; for example, the t and r restrictions above cut down the parameter space by a factor of 4 over the obvious t and r restrictions.

Lemma 3.11. *If $\alpha = (l + it, d + ib, r + ia) \in \mathcal{T}'$ has $d \leq \ln(3)$ and corresponds to a 2-generator group $\{G_\alpha, f_\alpha, w_\alpha\}$, then there exists a parameter $\beta = (l' + it', d' + ib', r' + ia') \in \mathcal{T}$ with associated group $\{G_\beta, f_\beta, w_\beta\}$ such that G_β is conjugate (in Isom($\mathbf{H}^3$)) to G_α.*

Proof. We note that $d < l/2$ is eliminated from consideration by Proposition 3.9i) and the definition of $\mathcal{T}'$. If $d \leq \ln(3)$, then $0.0978 \leq l \leq 1.289785$ by Propositions 3.8 and 3.9 ii) If for the marked group $\{G, f, w\}$ we have $-l/2 < r < 0$, then the marked group $\{G, f, w^{-1}\}$ is conjugate to an element of $\mathcal{T}$ whose new r-parameter is $-r$. Thus we can assume that conditions a, b, c from Definition 3.10 hold for the relevant $\{G, f, w\}$. Further, conditions e and f hold.

This leaves condition 3.10 d. Conjugating G by a reflection in the geodesic plane spanned by $B_{(0;\infty)}$ and $B_{(-1;1)}$ changes the t-parameter to $-t$ ($mod\ 2\pi$) but leaves the r and d parameters unchanged. The effect on b and a is irrelevant. □

By [G] Lemma 5.9 a closed orientable hyperbolic 3-manifold N satisfies the insulator condition provided that tuberadius(δ) $> \ln(3)/2$ for some closed geodesic $\delta \subset N$. Thus we are led to consider:

Problem 3.12. List all closed orientable hyperbolic 3-manifolds N possessing a shortest geodesic δ such that tuberadius(δ) $\leq \ln(3)/2$.

Remark 3.13. If a shortest geodesic δ in N satisfies tuberadius$(\delta) \leq \ln(3)/2$, then N gives rise to an element $\alpha \in \mathcal{T}$. (In fact N may give rise to finitely many different elements of $\mathcal{T}$.) Thus we need to investigate:

Problem 3.14. Find all parameters $\alpha = (l + it, d + ib, r + ia) \in \mathcal{T}$.

Remark 3.15. In the next paragraphs, we will describe our method of (partially) answering Problem 3.14. But before starting this description, we mention a technical point: starting with Definition 3.20, we will realize major advantages by working in the space $\mathcal{W} \supset \exp(\mathcal{P})$, and our results will ultimately be described in terms of $\mathcal{W}$. But for now, for simplicity, we will describe the results in terms of the unexponentiated space $\mathcal{P}$.

We will partition $\mathcal{P}$ into about one billion regions $\{\mathcal{P}_i\}$ and show that $\mathcal{T}$ is disjoint from all but seven small such regions. Suppose that $\mathcal{P}_i$ is a region of this partition and $\alpha \in \mathcal{P}_i$. Let h be a word in the letters f, w and their inverses. Associated to the parameter $\alpha = (l_\alpha + it_\alpha, d_\alpha + ib_\alpha, r_\alpha + ia_\alpha)$ there are the group elements f_α, w_α and hence h_α. If h_α is not the identity then we ask

a) Is R$elength(h_\alpha) <$ R$elength(f_\alpha) = l_\alpha$?

b) Is Redistance$(h_\alpha(B_{(0;\infty)}), B_{(0;\infty)})$ $<$ Redistance$(w_\alpha(B_{(0;\infty)}), B_{(0;\infty)})$ $= d_\alpha$?

If either a) or b) is true, then $\alpha \notin \mathcal{T}$.

Now let $\beta \in \mathcal{P}_i$, with f_β, w_β, and h_β the associated hyperbolic isometries. If say a) is true for α then so is the statement R$elength(h_\beta)$ $<$ R$elength(f_\beta)$ $= l_\beta$ for β sufficiently close to α. Thus we can show that $\mathcal{T} \cap \mathcal{P}_i = \emptyset$ if we can find an α for which, say, statement a) is true, and then use first-order Taylor approximation (with error/remainder term) to show that the corresponding statement holds for all $\beta \in \mathcal{P}_i$ while continuing to avoid the prohibition that h_β not be the identity.

Definition 3.16. A word h in w, f, w^{-1}, f^{-1} for which statement a) (resp. b)) in Remark 3.15 holds for each $\beta \in \mathcal{P}_i$ and for which h_β is not a power of f_β for each $\beta \in \mathcal{P}_i$ is called a *k*illerword for $\mathcal{P}_i$ with respect to contradiction a) (resp. b)).

Summary 3.17. With seven exceptions, to each of the approximately one billion regions partitioning $\mathcal{P}$, we will associate a killerword and a contradiction.

Remark 3.18. Computers are well suited for partitioning a set such as $\mathcal{P}$ into many regions $\{\mathcal{P}_i\}$, and finding a killerword h_i which eliminates all $\alpha_i \in \mathcal{P}_i$ due to contradiction C_i. Depending on the contradiction, we find computable expressions for approximations of the values of R$elength(h_\beta)$ or Redistance$(h_\beta(B_{(0;\infty)}), B_{(0;\infty)})$ and thus use the computer to eliminate all of $\mathcal{P}_i$.

Remark 3.19. . To analyze Relength and Redistance as in Remark 3.15, we would be led to work with the A*rccosh* function, because,

$$l\mathit{ength}(f) = 2\mathrm{A}\mathit{rccosh}(t\mathit{race}(A)/2)$$

where $A \in \mathrm{S}L(2, \mathbf{C})$ represents the isometry f. (As we do not need this formula for l*ength*(f) we will neither prove the formula, nor explain technical details about it.) This would be problematic from the view-point of error analysis—we do not want to deal with transcendental functions such as A*rccosh*.

This problem can be avoided slickly by exponentiating the preliminary parameter space $\mathcal{P}$ to get the parameter space $\mathcal{W}$ (the definition of $\mathcal{W}$ is given in Definition 3.20). Lemmas 3.22 and 3.23 then demonstrate that while working in $\mathcal{W}$ one need only understand the basic arithmetic operations $+, -, \times, /, \sqrt{}$. The machine implementation of these basic operations is governed by the IEEE standard IEEE-754 (see [IEEE]).

To expand a bit on the problematic nature of transcendental functions, we note that our computer version of Taylor approximations (see Preview 3.32 and Chapters 5 and 7) is designed to work for functions built up out of the basic arithmetic operations. It would be a nightmare to include functions such as the Arccosh.

Definition 3.20. Let

$$\mathcal{W} = \{(x_0, x_1, x_2, x_3, x_4, x_5) : |x_i| \leq 4 \times 2^{(5-i)/6} \; for \; i = 0, 1, 2, 3, 4, 5\}$$

$$\supset \exp(\mathcal{P}) = \{(x_0, x_1, x_2, x_3, x_4, x_5) \mid x_0 + ix_3 = \exp(e), \;\; x_1 + ix_4 = \exp(f), \\ x_2 + ix_5 = \exp(g) \; w\mathit{here}(e, f, g) \in \mathcal{P}\}$$

and let

$$\mathcal{S} = \exp(\mathcal{T}).$$

As we are taking exp of the various complex co-ordinates, it is notationally convenient to replace our complex parameters $L = l + it, \; D = d + ib, \; R = r + ia$ by exponentiated versions. That is, let

$$L' = \exp(L) = \exp(l + it), \;\; D' = \exp(D) = \exp(d + ib),$$
$$R' = \exp(R) = \exp(r + ia).$$

Remarks 3.21. i) We work with $\mathcal{W}$ instead of $\exp(\mathcal{P})$ because we want our initial parameter space to be a (6-dimensional) box that is easily subdivided. This has the side effect that certain regions (sub-boxes) $\mathcal{W}_i$ of $\mathcal{W}$ will be eliminated because they are outside of $\exp(\mathcal{P})$ not because of the analogues of conditions a) and b) in Remark 3.15 The entire collection of conditions is given in Chapter 4.

ii) The presence of the factor $2^{(5-i)/6}$ in the definition of $\mathcal{W}$ is explained in Construction 4.3. Briefly, the main reason for including it is to make the shape of regions stay as uniform

and "round" as possible under subdivision. This makes the Taylor approximations efficient, hence fast.

iii) We chose the co-ordinates of $\mathcal{W}$ so that $L' = x_0 + ix_3,\ D' = x_1 + ix_4,\ R' = x_2 + ix_5$ to gain a mild computer advantage.

Lemma 3.22. *If $(L', D', R') \in \mathcal{W}$ and $\{G, f, w\}$ is the associated normalized marked group, then f and w have matrix representatives*

a) $$f = \begin{pmatrix} \sqrt{L'} & 0 \\ 0 & 1/\sqrt{L'} \end{pmatrix},$$

b) $$w = \begin{pmatrix} \sqrt{R'} * ch & \sqrt{R'} * sh \\ sh/\sqrt{R'} & ch/\sqrt{R'} \end{pmatrix}$$

where $ch = (\sqrt{D'} + 1/\sqrt{D'})/2$ *and* $sh = (\sqrt{D'} - 1/\sqrt{D'})/2$.

Proof. a) In our set-up the (oriented) axis of f is $B_{(0;\infty)}$. As such, f corresponds to a diagonal matrix, with diagonal entries p and p^{-1}, with $|p| > 1$. The action of f on the bounding complex plane is simply multiplication by p^2. Extending this action to upper-half-space in the natural way rotates the z-axis by angle $\arg(p^2)$ and sends $(0, 0, 1)$ to $(0, 0, |p|^2)$. Thus,

$$\mathrm{I}m(\mathrm{l}ength(f)) = \arg(p^2) = \mathrm{I}m(\ln(p^2))$$

and, using the hyperbolic metric,

$$\mathrm{R}e(\mathrm{l}ength(f)) = \ln(|p|^2) = \mathrm{R}e(\ln(p^2)).$$

That is, $\mathrm{l}ength(f) = \ln(p^2)$ and

$$p = \pm\exp(\mathrm{l}ength(f)/2) = \pm\sqrt{\exp(\mathrm{l}ength(f))} = \pm\sqrt{\exp(L)} = \pm\sqrt{L'}.$$

Now, we take the positive square root (taking the negative square root produces the other lift from $PSL(2,\mathbf{C})$ to $SL(2,\mathbf{C})$).

b) $w = \beta \circ \alpha$ where β is translation of distance R along $B_{(0;\infty)}$ and α is translation of distance D along $B_{(-1;1)}$. Thus, a matrix representative of β is

$$\begin{pmatrix} \sqrt{R'} & 0 \\ 0 & 1/\sqrt{R'} \end{pmatrix}$$

and a matrix representative of α can be computed to be

$$\begin{pmatrix} \cosh(D/2) & \sinh(D/2) \\ \sinh(D/2) & \cosh(D/2) \end{pmatrix}.$$

But $\cosh(D/2) = (\exp(D/2) + \exp(-D/2))/2 = (\sqrt{D'} + 1/\sqrt{D'})/2 = ch$ and similarly for sh. Thus,

$$\alpha = \begin{pmatrix} ch & sh \\ sh & ch \end{pmatrix}$$

and b) follows by matrix multiplication. □

Lemma 3.23. *If $h \in Isom_+(\mathbf{H}^3)$ is represented by the matrix*

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{S}L(2, \mathbf{C}),$$

then

- *a)* $\exp(Relength(h)) = |trace(A)/2 \pm \sqrt{(trace(A)/2)^2 - 1}|^2$,
- *b)* $\exp(Redistance(h(B_{(0;\infty)}), B_{(0;\infty)})) = |orthotrace(A) \pm \sqrt{(orthotrace(A))^2 - 1}|$ *where* $orthotrace(A) = ad + bc$.

In both cases, the $+, -$ produce reciprocal values for the right-hand side, and we take the one producing the larger value, unless the value is 1*, in which case there is no need to choose.*

Proof. a) If A is elliptic or parabolic, the proof is straightforward (the trace of a parabolic is ± 2 while the trace of an elliptic is a real number between 2 and -2).

We assume A is hyperbolic. Because trace is a conjugacy invariant, we can assume the oriented axis of A is $B_{(0;\infty)}$. Thus A is a diagonal matrix with p and p^{-1} along the diagonal with $|p| > 1$, and, as in the proof of Lemma 3.22, we see that $\exp(length(h)) = p^2$. Of course, $trace(A) = p + p^{-1}$, and it is easy enough to solve for p. Specifically, $p = trace(A)/2 \pm \sqrt{(trace(A)/2)^2 - 1}$. Thus,

$$\begin{aligned} \exp(\mathrm{R}elength(h)) &= |\exp(\mathrm{l}ength(h))| = |p|^2 \\ &= |(trace(A)/2) \pm \sqrt{(trace(A)/2)^2 - 1}|^2. \end{aligned}$$

b) If $B_{(0;\infty)}$ and $h(B_{(0;\infty)})$ intersect at infinity, then the proof is straightforward. For example, if h fixes the point $(0, 0, 0)$ at infinity, then $c = 0$, $ad = 1$ and the formula holds. Similarly for the other cases in which $B_{(0;\infty)}$ and $h(B_{(0;\infty)})$ intersect at infinity.

We assume $B_{(0;\infty)}$ and $h(B_{(0;\infty)})$ do not intersect at infinity. We will compute the length of k, the square of the transformation taking $B_{(0;\infty)}$ to $h(B_{(0;\infty)})$ along their ortholine. Let τ be 180-degree rotation about $B_{(0;\infty)}$, then $(h \circ \tau \circ h^{-1})$ is 180-degree rotation about $h(B_{(0;\infty)})$, and we have that $k = (h \circ \tau \circ h^{-1}) \circ \tau$. Now, τ and h are represented by the matrices

$$\begin{pmatrix} i & 0 \\ 0 & -i \end{pmatrix} \quad and \quad \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{S}L(2, \mathbf{C}).$$

Hence, $k = (h \circ \tau \circ h^{-1}) \circ \tau$ can be computed to have matrix representation

$$\begin{pmatrix} ad+bc & 2ab \\ 2cd & ad+bc \end{pmatrix}.$$

Thus,

$$\begin{aligned} &\exp(\mathrm{R}edistance(h(B_{(0;\infty)}), B_{(0;\infty)})) \\ &\quad = \exp(\mathrm{R}elength(k)/2) = \sqrt{|\exp(\mathrm{l}ength(k)|} \\ &\quad = |(trace(k)/2) \pm \sqrt{(trace(k)/2)^2 - 1}| \\ &\quad = |(ad+bc) \pm \sqrt{(ad+bc)^2 - 1}|. \end{aligned}$$

□

Remark 3.24. i) It follows from Lemma 3.23 that if h is a word in f, w, f^{-1}, w^{-1}, then for any parameter value $\alpha \in \mathcal{W}$,

$$\exp(\mathrm{R}elength(h_\alpha)), \text{ and } \exp(\mathrm{R}edistance(h_\alpha(B_{(0;\infty)}), B_{(0;\infty)}))$$

can be computed using only the operations $+, -, \times, /, \sqrt{}$.

ii) During the course of the computer work needed to prove the main theorems, the parameter space $\mathcal{W}$ was decomposed into sub-boxes by computer via a recursive subdivision process: Given a sub-box being analyzed, either it can be *k*illed directly (that is, eliminated by a killerword and associated condition as described in Remark 3.15 or for the trivial reason described in Remark 3.21 i), or it cannot. If it cannot be killed directly, it is subdivided in half by a hyperplane $\{x_i = c\}$ (where i runs through the various co-ordinate dimensions cyclically) and the two pieces are analyzed separately, and so on.

As such, a sub-box of $\mathcal{W}$ can be described by a sequence of 0's and 1's where 0 means "take the lesser x_i values" and 1 means "take the greater x_i values." For the decomposition of $\mathcal{W}$ into sub-boxes, all the sub-box descriptions could be neatly encoded into one tree (although in practice we found it preferable to use several trees to describe the entire decomposition. See Chapter 4).

iii) In the following proposition, seven *e*xceptional boxes are described as sequences of 0's and 1's. Four of the exceptional boxes—X_0, X_4, X_5, X_6—are each the union of two abutting sub-boxes, $X_0 = X_{0a} \cup X_{0b}$ and so on. It is a pleasant exercise to work through the fact that they abut. It should be noted that had the set-up for $\mathcal{W}$ been different, more sub-boxes (or perhaps fewer) might have been needed to construct the seven exceptional regions.

It is also a pleasant exercise to calculate by hand the co-ordinate ranges of the various sub-boxes. For example, the range of the last co-ordinate (i.e., x_5) of the sub-box

$X_{6a} = 111000000001000111\ 111111110101001111\ 011111010111111111$
$110001001011000111\ 0$

is found by taking the 6th entry, the 12th entry, the 18th entry, and so on. These entries are 011111111111. The first entry (0) means take the lesser x_5 values, and produces the interval $[-4, 0]$. The second entry (1) means take the greater x_5 values, and produces the interval $[-2, 0]$. The third entry (1) produces $[-1, 0]$. Continuing, we see that X_{6a} has $-2^{-9} \leq x_5 = \mathrm{I}m(R') \leq 0$. The other co-ordinates can be computed in the same fashion, although they must at the end be multiplied by the factor $2^{(5-i)/6}$ (see the definition of the initial box $\mathcal{W}$). The range of co-ordinate values for each exceptional box $X_0, X_1, \ldots, X_6$ is given in Table 3 (a limited number of significant digits is given), and then a range of co-ordinates for exceptional regions (in $\mathcal{P}$) $\mathcal{R}_i \supset \exp^{-1}(X_i)$ is given (see Remarks 3.28i) and 3.28ii) in Table 3. (Note that this use of the symbol $\mathcal{R}_i$ differs slightly from the use in §0.) Finally, two *q*uasi-relators are given in Theorem 2.1 for each exceptional box $X_0, X_1, \ldots, X_6$ (see the next definition).

Definition 3.25. A *q*uasi-relator in a sub-box X of $\mathcal{W}$ is a word in $f, w, F = f^{-1}, W = w^{-1}$ that is close to the identity throughout X and experimentally appears to be converging to the identity at some point in X. In particular, a quasi-relator rigorously has Relength less than that of f at all points in X.

Theorem 3.26. *2.1 Within the parameter space $\mathcal{W}$ but outside the seven exceptional boxes there are no parameter points corresponding to marked groups $\{G, f, w\}$ where G is discrete, torsion-free and parabolic-free; f corresponds to a shortest geodesic δ of tuberadius $\leq \ln(3)/2$; and w takes a particular lift of δ to a nearest translate. Specifically, $\mathcal{S} \cap (\mathcal{W} - \bigcup_{n=0,\ldots,6} X_n) = \emptyset$ where the X_n are the exceptional boxes*

$X_0 = X_{0a} \cup X_{0b}$,

$X_{0a} = 001000110111110001\ 101001010101011001\ 011011010111101101$
$100001101101000111\ 010001110101100101\ 1101110111110100,$

$X_{0b} = 001001110110110000\ 101000010100011000\ 011010010110101100$
$100000101100000110\ 010000110100100100\ 1101100111100100,$

X_0 *quasi-relators:*
$r_1 = fwFwwFwfww,$
$r_2 = FwfwfWfwfw,$

$X_1 = 001000110001110110\ 011101000110111110\ 100010110000100011$
$101101001101001000\ 110101011000000100\ 000.$

X_1 *quasi-relators:*
$r_1 = FFwFWFWfWFWFwFFww,$
$r_2 = FFwwFwfwfWfwfwFww,$

$X_2 = 001000110101010010\ 101010110001100101\ 110111100001101010$

111100100000010001 111100,

X_2 *quasi-relators:*
$r_1 = FwfwfWffWfwfwFww$,
$r_2 = FFwFFwwFwfwfwFww$,

$X_3 = 111000000001000110\ 011011101101011000\ 111101011110001100$
$111111100110110000\ 0000100010100010$,

X_3 *quasi-relators:*
$r_1 = FFwfwFFwwFWFwFWfWFWffWFWfWFwFWFww$,
$r_2 = FFwfwFwfWfwfWWfwfWfwFwfwFFwwFWFww$,

$X_4 = X_{4a} \cup X_{4b}$,

$X_{4a} = 111000000001000110\ 011001001111101010\ 011110110110111101$
$100011111110110110\ 10000111101$,

$X_{4b} = 111000000001000110\ 011001001111101010\ 111110010110011101$
$000011011110010110\ 00000101101$,

X_4 *quasi-relators:*
$r_1 = FFwfwFwfWfwfWfwFwfwFFwwFWFwFWFww$,
$r_2 = FFwfwFwfwFFwwFWFwFWfWFWfWFwFWFww$,

$X_5 = X_{5a} \cup X_{5b}$,

$X_{5a} = 001000110001110111\ 001111000101111111\ 101111100111001111$
$000001111011110111\ 1$,

$X_{5b} = 001001110000110110\ 001110000100111110\ 101110100110001110$
$000000111010110110\ 1$,

X_5 *quasi-relators:*
$r_1 = FwFWFwFwfwfWfwfw$,
$r_2 = FwfwfWfWFWfWfwfw$,

$X_6 = X_{6a} \cup X_{6b}$,

$X_{6a} = 111000000001000111\ 111111110101001111\ 011111010111111111$
$110001001011000111\ 0$,

$X_{6b} = 111001000000000110\ 111110110100001110\ 011110010110111110$
$110000001010000110\ 0$,

X_6 *quasi-relators:*
$r_1 = FWFwFWfWFwFWFwfw$,
$r_2 = FWFwfwFwfWfwFwfw$.

Proof. The proof follows along the lines presented in Remark 3.15 Two computer files contain the data needed for the proof. The first computer file describes the partition of $\mathcal{W}$ into sub-boxes and attaches an integer to each such sub-box, and the second file, called "conditionlist" is an ordered list of conditions and killerwords. The integer associated to a sub-box in the first file describes the numbered condition/killerword from conditionlist that will eliminate the sub-box in question (other than those corresponding to the X_i). A computer program named *v*erify shows that the conditions and killerwords in question actually do kill off their associated sub-boxes (see Section 4 for more details). This computer program addresses the issues of Remark 3.18. The code for *v*erify is available at the *A*nnals web site.

In addition, a mild modification of *v*erify showed that the listed words were quasi-relators for the given sub-boxes. □

Corollary 3.27. *If δ is a shortest geodesic in N, a closed orientable hyperbolic 3-manifold, then*

- *i) either tuberadius(δ) $>$ $\ln(3)/2$ or $\exp(length(\delta)) \in \mathcal{L}(X_k)$ for some $k \in 0, \ldots, 6$ where $\mathcal{L}(X_k)$ denotes the range of L' values in the exceptional box X_k.*
- *ii) Either tuberadius(δ) $>$ $\ln(3)/2$ or tuberadius(δ) $=$ $\mathrm{Re}(D)/2$ where $\exp(D) \in \mathcal{D}(X_k)$ for some $k \in 0, \ldots, 6$ and $\mathcal{D}(X_k)$ denotes the range of D' values in the exceptional box X_k.* □

Remarks 3.28. i) The values in Table 1.1 are only approximations of actual values which can be computed as in Remark 3.24iii.

ii) The values in Table 3 correspond to boxes in $\mathcal{P}$ which contain the natural log of the (exceptional) boxes in Table 3 (here we use the true co-ordinates of the boxes, not just the approximation-by-truncation co-ordinates). For example, the rectangle in $\mathbf{C}$ determined by $l_{min}, l_{max},\ t_{\min}, t_{max}$ for $\mathcal{R}_0$ contains the natural log of the rectangle in $\mathbf{C}$ determined by $l'_{\min}, l'_{max}, t'_{\min}, t'_{max}$ for X_0.

The following conjecture apppeared in [GMT]. It has since been proven, with item iii) amended; see the Introduction

Conjecture 3.29. *Each exceptional box $X_i, 0 \leq i \leq 6$, contains a unique element s_i of S. Further, if $\{G_i, f_i, w_i\}$ is the marked group associated to s_i then $N_i = \mathbf{H}^3/G_i$ is a closed hyperbolic 3-manifold with the following properties:*

- *i) N_i has fundamental group $\langle f, w; r_1(X_i), r_2(X_i)\rangle$, where $r_1(X_i),\ r_2(X_i)$ are the quasi-relators associated to the box X_i.*
- *ii) N_i has a Heegaard genus 2 splitting realizing the above group presentation.*
- *iii) N_i nontrivially covers no manifold.*

Table 3.1: Exceptional boxes in (L', D', R') co-ordinates in $\mathcal{W}$; truncated values

X_0

$l'_{min} = -0.84065$	$l'_{max} = -0.84060$	$t'_{min} = -2.13726$	$t'_{max} = -2.13722$
$d'_{min} = -0.84064$	$d'_{max} = -0.84059$	$b'_{min} = -2.13729$	$b'_{max} = -2.13722$
$r'_{min} = 0.999979$	$r'_{max} = 1.000022$	$a'_{min} = -0.00006103$	$a'_{max} = 0.00006103$

X_1

$l'_{min} = -1.34852$	$l'_{max} = -1.34831$	$t'_{min} = -2.66102$	$t'_{max} = -2.66072$
$d'_{min} = -0.54334$	$d'_{max} = -0.54315$	$b'_{min} = -2.85877$	$b'_{max} = -2.85849$
$r'_{min} = 0.90390$	$r'_{max} = 0.90408$	$a'_{min} = -1.47167$	$a'_{max} = -1.47143$

X_2

$l'_{min} = -1.78701$	$l'_{max} = -1.78527$	$t'_{min} = -2.27253$	$t'_{max} = -2.27130$
$d'_{min} = -1.07428$	$d'_{max} = -1.07273$	$b'_{min} = -2.71846$	$b'_{max} = -2.71736$
$r'_{min} = 0.74163$	$r'_{max} = 0.74301$	$a'_{min} = -1.52929$	$a'_{max} = -1.52832$

X_3

$l'_{min} = 0.58117$	$l'_{max} = 0.58160$	$t'_{min} = -3.31221$	$t'_{max} = -3.31190$
$d'_{min} = 1.15644$	$d'_{max} = 1.15683$	$b'_{min} = -2.75628$	$b'_{max} = -2.75573$
$r'_{min} = 1.40420$	$r'_{max} = 1.40454$	$a'_{min} = -1.17968$	$a'_{max} = -1.17919$

X_4

$l'_{min} = 0.33321$	$l'_{max} = 0.33495$	$t'_{min} = -3.31959$	$t'_{max} = -3.31898$
$d'_{min} = 0.97739$	$d'_{max} = 0.97817$	$b'_{min} = -2.82533$	$b'_{max} = -2.82478$
$r'_{min} = 1.35413$	$r'_{max} = 1.35482$	$a'_{min} = -1.22558$	$a'_{max} = -1.22460$

X_5

$l'_{min} = -1.37984$	$l'_{max} = -1.37810$	$t'_{min} = -2.53706$	$t'_{max} = -2.53460$
$d'_{min} = -1.37967$	$d'_{max} = -1.37657$	$b'_{min} = -2.53650$	$b'_{max} = -2.53430$
$r'_{min} = 0.99989$	$r'_{max} = 1.00265$	$a'_{min} = -0.001953$	$a'_{max} = 0.001953$

X_6

$l'_{min} = 1.37810$	$l'_{max} = 1.37984$	$t'_{min} = -2.53706$	$t'_{max} = -2.53460$
$d'_{min} = 1.37657$	$d'_{max} = 1.37967$	$b'_{min} = -2.53650$	$b'_{max} = -2.53430$
$r'_{min} = 0.99989$	$r'_{max} = 1.00265$	$a'_{min} = -0.001953$	$a'_{max} = 0.001953$

Table 3.2: Exceptional regions (boxes) in (L, D, R) co-ordinates in $\mathcal{P}$

$\mathcal{R}_0$

$l_{min} = 0.8314$	$l_{max} = 0.8315$
$t_{min} = -1.9456$	$t_{max} = -1.9455$
$d_{min} = 0.8314$	$d_{max} = 0.8315$
$b_{min} = -1.9456$	$b_{max} = -1.9455$
$r_{min} = -0.00002051$	$r_{max} = 0.00002267$
$a_{min} = -0.00006105$	$a_{max} = 0.00006105$

$\mathcal{R}_1$

$l_{min} = 1.0928$	$l_{max} = 1.0931$	$t_{min} = -2.0399$	$t_{max} = -2.0397$
$d_{min} = 1.0680$	$d_{max} = 1.0682$	$b_{min} = -1.7587$	$b_{max} = -1.7585$
$r_{min} = 0.5463$	$r_{max} = 0.5465$	$a_{min} = -1.0201$	$a_{max} = -1.0198$

$\mathcal{R}_2$

$l_{min} = 1.0608$	$l_{max} = 1.0617$	$t_{min} = -2.2375$	$t_{max} = -2.2366$
$d_{min} = 1.0720$	$d_{max} = 1.0727$	$b_{min} = -1.9473$	$b_{max} = -1.9466$
$r_{min} = 0.5298$	$r_{max} = 0.5308$	$a_{min} = -1.1193$	$a_{max} = -1.1182$

$\mathcal{R}_3$

$l_{min} = 1.2126$	$l_{max} = 1.2129$	$t_{min} = -1.3972$	$t_{max} = -1.3969$
$d_{min} = 1.0947$	$d_{max} = 1.0951$	$b_{min} = -1.1736$	$b_{max} = -1.1733$
$r_{min} = 0.6063$	$r_{max} = 0.6067$	$a_{min} = -0.6988$	$a_{max} = -0.6984$

$\mathcal{R}_4$

$l_{min} = 1.2046$	$l_{max} = 1.2050$	$t_{min} = -1.4708$	$t_{max} = -1.4702$
$d_{min} = 1.0949$	$d_{max} = 1.0953$	$b_{min} = -1.2378$	$b_{max} = -1.2374$
$r_{min} = 0.6019$	$r_{max} = 0.6027$	$a_{min} = -0.7357$	$a_{max} = -0.7349$

$\mathcal{R}_5$

$l_{min} = 1.0595$	$l_{max} = 1.0606$	$t_{min} = -2.0694$	$t_{max} = -2.0683$
$d_{min} = 1.0591$	$d_{max} = 1.0604$	$b_{min} = -2.0694$	$b_{max} = -2.0680$
$r_{min} = -0.0001069$	$r_{max} = 0.002654$	$a_{min} = -0.001954$	$a_{max} = 0.001954$

$\mathcal{R}_6$

$l_{min} = 1.0595$	$l_{max} = 1.0606$	$t_{min} = -1.0733$	$t_{max} = -1.0722$
$d_{min} = 1.0591$	$d_{max} = 1.0604$	$b_{min} = -1.0736$	$b_{max} = -1.0722$
$r_{min} = -0.0001069$	$r_{max} = 0.002654$	$a_{min} = -0.001954$	$a_{max} = 0.001954$

- *iv)* N_6 *is isometric to* N_5.
- *v) If* (L_i, D_i, R_i) *is the parameter in* $\mathcal{T}$ *corresponding to* s_i*, then* L_i, D_i, R_i *are related as follows.*

$$\text{For } X_0,\ X_5,\ X_6: \quad L = D, \quad R = 0.$$
$$\text{For } X_1,\ X_2,\ X_3,\ X_4: \quad R = L/2.$$

The following conjecture is a succinct, though slightly weaker form of Conjecture 3.29.

Conjecture 3.30. *If* δ *is a shortest geodesic in a closed orientable hyperbolic* 3*-manifold* N*, then either tuberadius*$(\delta) > \ln(3)/2$ *or* N *is one of six exceptional manifolds.*

Remark 3.31. Here we outline our method for finding a decomposition of the initial box $\mathcal{W}$ into sub-boxes (other than those making up the seven exceptional boxes) each with a condition/killerword that kills off the entire associated sub-box. For convenience, we will generally refer to a "sub-box" simply as a "box."

A simple algorithm for finding a killerword for a region is as follows. Work with a set of words to consider, initialized to the null word. At each step, remove the oldest word from the set, and test to see if that word is a killerword. If it is not, put the word back into the set, concatenated with each of the generators and their inverses. Eventually, this algorithm will enumerate all words, and so, if there is a killerword, the algorithm will eventually find it. In practice, there are two problems with this approach: there is no provision for the possibility that no killerword exists for the region under consideration, and the time to find a word of length n grows exponentially.

When we also take into account the possibility of subdividing the box, getting an answer in finite time will be possible; but the search is in practice very expensive. The most obvious way of speeding it up is to avoid the search entirely when feasible: a killerword works on a neighborhood of a region, and by testing killerwords found for nearby boxes, most of the time the search is not necessary.

Still, there are words of length as long as 44 that were considered, and testing all of the roughly 3^{44} combinations would be prohibitive on today's computers. In practice (due to a bug), the search algorithm used for most of the parameter space was no better than the brute-force method just described, but to find killerwords for the remaining regions, an improvement was needed. Rather than blindly selecting words in first-in-first-out order, the algorithm can rank the words under consideration based on a heuristic estimate of the likelihood of their being useful (a word is *u*seful if it is a prefix of a killerword). We note first that short words tend to be better than long words, as they have fewer steps and less error. Second, we note that words with a large translation distance are given a bad ranking, for two reasons: they will need more generators appended before they get back to the small translation distance which is needed for a contradiction, and computations with those words introduce more error per step than computations with closer words.

This approach was an improvement, but was not finding enough killerwords in the regions around X_3 and X_4. Further investigation showed that the algorithm was getting stuck on an identity: once it found an identity, it would consider only words which started with that identity, and ignored all of the other words. To fix this problem, a "diversity" heuristic was introduced, to give special consideration to unlikely but unusual words.

To prevent the search from running forever, it is temporarily abandoned after some number of steps, and re-done with twice as many steps every time the number of descendant boxes doubles. This way, the search could run forever, but only if the subdivision process runs forever. This merged process of alternately searching and subdividing we call the *d*ecomposition algorithm.

The decomposition algorithm went through several revisions; at each stage of the revision process, the algorithm effectively increased the extent to which killerwords found for one region were used to kill other regions. The first attempt—used to determine the feasibility of the whole effort— iterated over regions in depth-first order, performing the search as described above. At that stage, it became evident that the search process, as opposed to the subdivision process, was consuming nearly all of the computation time, and so the second version iterated over regions in breadth-first order, and, once it found a killerword, tried to use that word on all adjacent regions.

The breadth-first version was used to analyze the entire parameter space, although it skipped some parts due to various bugs; the search heuristic was replaced once, and there was considerable human input to tell the search about particularly difficult killerwords, or to tweak its search parameters (length, and weightings in the heuristic).

The third stage of the revision process reduced the number of boxes by attempting all found killerwords in a large region (about a thousand boxes) on all boxes in the region. It did not do any searching, since it was provided with a list of killerwords known to work.

The final version was created when the bugs in evaluation were brought to light, and the existing killerword tree was found to be insufficient. It used the list of killerwords used for the entire tree, and some statistics about the number of subdivisions required in order for a given word to kill a particular box, and evaluated each word on each box. Whenever a word was evaluated, a kind of triage was used to determine whether that word was likely to kill the box in question, likely to kill any of its n^{th} generation descendants, or unlikely to kill any descendants of the box; the answer to that heuristic either allowed more detailed evaluation (with the error term included), deferred further evaluation until the box had been subdivided n more times, or excluded that word from further consideration on any descendant of the box. With these heuristics, this program wound up evaluating on average about 10 of the roughly 13200 words per box, and was able to construct the tree consisting of the decomposition into sub-boxes with associated conditions/killerwords.

We mention that the bugs, complexity, and frequent changes in the search programs are irrelevant to the accuracy of the verification. In fact, we shielded the verification programs from internal issues related to the searching programs. Given a putative decomposition of the parameter space into sub-boxes with associated conditions/killerwords the program *v*erify simply checks whether this decomposition with conditions/ killerwords works.

Preview 3.32. In Remark 3.15, we mentioned that we use first-order Taylor approximations, with remainder term, to show that a killerword which eliminates a point $x \in \mathcal{W}_i$ eliminates all of $\mathcal{W}_i$. The computational object we constructed to carry out these Taylor approximations is called an *A*ffApprox. In the parameter space $\mathcal{W}$, all functions analyzed via Taylor approximations in this way are built up from the operations $+$, $-$, $\times$, $/$, $\sqrt{\cdot}$. We prove combination formulas for these operations, which show how the Taylor approximations (including the remainder term) change when one of these operations is applied to two AffApproxes. This is carried out in Chapter 5

To ensure that all of our computer calculations are rigorous, we use a round-off error analysis. Typically, this is done by using interval arithmetic on floating-point numbers. Instead, we introduce round-off error at the level of AffApproxes and incorporate the round-off error into the remainder term. The main reason for this additional complexity is to get more accuracy in our calculation of AffApproxes, which allows us to analyze substantially fewer boxes. Further, the individual computations are faster. This is all carried out in Sections 6 and 7.

Chapter 4

Conditions and sub-boxes

In this section we expand on some topics mentioned briefly in Chapter 3 . As such, it would be useful to look again at Definitions 3.6, 3.10, and 3.20, and Remark 3.7), where $\mathcal{P}$, $\mathcal{T}$ and their partners (under exponentiation) $\mathcal{W}$, $\mathcal{S}$ are introduced, and at Definition 3.16 where the notion of a killerword is introduced (the definition is phrased in terms of $\mathcal{P}$, but the definition also makes sense for $\mathcal{W}$). Note that working with the region $\mathcal{P}$ is intuitively appealing, but working with the box $\mathcal{W}$ is vastly superior computationally (see Remark 3.19).

The proof of Theorem 2.1 amounts to decomposing $\mathcal{W}$ into a collection of sub-boxes of two types:

1) The 11 sub-boxes which comprise the exceptional boxes $X_0, X_1, \ldots, X_6$.

2) Sub-boxes each of which has an associated *c*ondition that will describe how to kill that entire sub-box, perhaps with the help of a killerword. To *k*ill a sub-box means to show that $\mathcal{S} - \bigcup_{n=0,\ldots,6} X_n$ has no point in the sub-box.

The set-up for efficiently describing these sub-boxes will be given in Construction 4.3.

We now list the conditions used to kill the nonexceptional sub-boxes. There are two types of conditions: the trivial and the interesting. The trivial conditions kill sub-boxes in $\mathcal{W}$ since the sub-boxes in question miss $\exp(\mathcal{P})$. The interesting conditions are where the real work is done, and they require a killerword in f, w, f^{-1}, w^{-1} to work their magic (see Remark 3.15).

To be consistent with the computer program *v*erify we use the following notation: $L' = z_0 + iz_3$, $D' = z_1 + iz_4$, and $R' = z_2 + iz_5$. Here $(L', D', R') \in \mathcal{W}$ and $L' = \exp(L) = \exp(l + it)$, $D' = \exp(D) = \exp(d + ib)$, $R' = \exp(R) = \exp(r + ia)$.

The Trivial Conditions 4.1. *C*ondition 's' (*s*hort): Tests that all points in the sub-box have $|z_0 + iz_3| < 1.10274$. This ensures that

$$\exp(l) = |\exp(L)| = |L'| = |z_0 + iz_3| < 1.10274 < \exp(0.0978),$$

and Definition 3.10 tells us that we are outside of $\exp(\mathcal{P})$.

*C*ondition 'l' (*l*ong): Tests that all points in the sub-box have $|z_0 + iz_3| > 3.63201$. This ensures that

$$\exp(l) = |\exp(L)| = |L'| = |z_0 + iz_3| > 3.63201 > \exp(1.289785)$$

and we are outside of $\exp(\mathcal{P})$.

*C*ondition 'n' (*n*ear): Tests that all points in the sub-box have $|z_1 + iz_4| < 1$. This ensures that

$$\exp(d) = |\exp(D)| = |D'| = |z_1 + iz_4| < 1 = \exp(0)$$

and we are outside of $\exp(\mathcal{P})$.

*C*ondition 'f' (*f*ar): Tests that all points in the sub-box have $|z_1 + iz_4| > 3$. This ensures that

$$\exp(d) = |\exp(D)| = |D'| = |z_1 + iz_4| > 3 = \exp(\ln 3)$$

and we are outside of $\exp(\mathcal{P})$.

*C*ondition 'w' (*w*hirle big): Tests that all points in the sub-box have $|z_2 + iz_5|^2 > |z_0 + iz_3|$ This ensures that

$$\exp(r) = |\exp(R)| = |R'| = |z_2 + iz_5| > \sqrt{|z_0 + iz_3|} = \sqrt{\exp(l)} = \exp(l/2)$$

and we are outside of $\exp(\mathcal{P})$.

*C*ondition 'W' (*w*hirle small): Tests that all points in the sub-box have $|z_2 + iz_5| < 1$. This ensures that

$$\exp(r) = |\exp(R)| = |R'| = |z_2 + iz_5| < 1 = \exp(0)$$

and we are outside of $\exp(\mathcal{P})$.

The Interesting Conditions 4.2. *C*ondition 'L': This condition comes equipped with a killerword k in f, w, f^{-1}, w^{-1}, and tests that all points in the sub-box have $|\exp(l\mathit{ength}(k))| < |L'| = |\exp(L)|$, where l*ength*$(k)$ means the length of the isometry determined by k. This, of course, contradicts the fact that L is the length of the shortest geodesic.

It is easy to carry out the test $|\exp(l\mathit{ength}(k))| < |L'|$ because Lemma 3.23 a) can be used. Note that in *v*erify the function which computes exp(l*ength*) is called *l*ength.

Of course, Condition 'L' also checks that the isometry corresponding to the word k is not the identity.

*C*ondition 'O': This condition comes equipped with a killerword k in f, w, f^{-1}, w^{-1}, and tests that all points in the sub-box have

$$|\exp(\mathrm{d}\mathit{istance}(k(B_{(0,\infty)}), B_{(0,\infty)}))| < |D'| = |\exp(D)|.$$

(Recall that $B_{(0;\infty)}$ denotes the oriented geodesic $\{(0,0,z) : 0 < z < \infty\}$, with negative endpoint $(0,0,0)$.) This, of course, contradicts the "nearest" condition.

It is easy to carry out the test $|\exp(\mathrm{d}istance(k(B_{(0,\infty)}), B_{(0,\infty)}))| < |D'|$ because Lemma 3.23 b) can be used. Note that in *v*erify the function which computes the quantity

$$\exp(\mathrm{d}istance(k(B_{(0,\infty)}), B_{(0,\infty)}))$$

is called *o*rthodist.

Also, Condition 'O' checks that the isometry corresponding to the word k does not take the axis of f to itself.

*C*ondition '2': This is just the 'L' condition without the "not-the-identity" check, but with the additional proviso that the killerword k is of the form $f^p w^q$. This ensures that k is not the identity, because for k to be the identity f and w would have to have the same axis, which contradicts the fact that d can be taken to be greater than or equal to $l/4$.

*C*ondition 'conjugate': There is one other condition that is used to eliminate points in $\mathcal{W}$. Following Definition 3.10 (and Lemma 3.11) we eliminate all boxes with $0 < t \leq \pi$. Of course, after exponentiating $L = l + it$, this corresponds to eliminating all boxes with $z_3 > 0$. Specifically, we toss all sub-boxes of $\mathcal{W}$ whose fourth entry is a 1. This condition does not appear in *v*erify because it is applied "outside" of these programs, as described in Construction 4.3.

Construction 4.3. : We now give the method for describing the roughly 930 million sub-boxes that the initial box $\mathcal{W}$ is subdivided into.

All sub-boxes are obtained by subdivision of a previous sub-box along a real hyper-plane midway between parallel faces of the sub-box before subdivision. Of course, these midway planes are of the form $x_i =$ a constant. We use 0's and 1's to describe which half of a subdivided sub-box to take (0 corresponds to lesser x_i values). For example, 0 describes the sub-box

$$\mathcal{W} \cap \{(x_0, x_1, x_2, x_3, x_4, x_5) : x_0 \leq 0\},$$

010 describes the sub-box

$$W \cap \{(x_0, x_1, x_2, x_3, x_4, x_5) : x_0 \leq 0,\ x_1 \geq 0,\ x_2 \leq 0\},$$

and so on.

In this way, we get a one-to-one correspondence between strings and sub-boxes. If s is a string of 0's and 1's, then let $Z(s)$ denote the sub-box corresponding to s. The range of values for the i^{th} coordinate in the sub-box $Z(s)$ is related to the binary fraction $0.s_i s_{i+6} \ldots s_{i+6k}$. The two sub-boxes gotten from subdividing $Z(s)$ are $Z(s0)$ and $Z(s1)$.

The directions of subdivision cycle among the various coordinate axes: the n^{th} subdivision is across the $(n \bmod 6)^{th}$ axis. The dimensions of the top-level box $\mathcal{W}$ were chosen so that subdivision is always done across the longest dimension of the box, and so that all of the

sub-boxes are similar. The dimensions of $\mathcal{W}$ have the beneficial effect of making the sub-boxes as "round" as possible, hence making the Taylor approximation calculations efficient and fast. This explains the factor of $2^{(5-i)/6}$ in Definition 3.20

To kill a sub-box $Z(s)$, the checker program has two (recursive) options: use a condition and, if necessary, an associated killerword to kill $Z(s)$ directly, or first kill $Z(s0)$ and then kill $Z(s1)$. At this point, it may seem as if the second option is not necessary, because surely a condition which kills two halves also kills the whole. The answer to this has been hinted at in Remarks 3.15 and 3.32 where it is noted that our evaluation of a function arising from a killerword is via first-order Taylor approximation, complete with remainder/error term. (Note that the remainder/error term incorporates bounds on both the theoretical error arising from using a first-order Taylor approximation to approximate a function, and the accumulated round-off error; see Chapters 5, 6, 7. Even if a killerword could theoretically kill off a sub-box, it is quite possible that our first-order Taylor approximation approach would not be able to prove this because its remainder/error term is too large. However, if we subdivide the sub-box, then the first-order Taylor approximations on the two halves should be more accurate. Thus, we want to have the recursive subdivision option at our disposal. Note that because the checker program does in fact do such recursive subdivisions, the actual number of sub-boxes in the ultimate subdivision is larger (perhaps substantially larger) than the 930 million sub-boxes of the initial data tree.

It is also possible that the checker program will employ neither of the two options described in the previous paragraph, and will instead employ a third option: do not kill $Z(s)$, and instead mark s as omitted. Any omitted sub-boxes are checked with another instance of the checker program, unless the sub-box is one of the 11 exceptional sub-boxes (which produce the seven exceptional boxes after joining abutters). Note that according to the definition of "kill" given at the beginning of this section, the exceptional boxes are automatically killed.

Thus, a typical output from *v*erify would be

$$\mathrm{v}\mathit{erified}0000001111011111111$$
$$-\{0000001111011111110 \;\; 000000111101111111110\}.$$

which means that the sub-box $Z(000000111101111111)$ was killed except for its sub-boxes $Z(0000001111011111110)$ and $Z(000000111101111111110)$. The output

$$\mathrm{v}\mathit{erified}0000001111011111110 - \{\ \}.$$

and

$$\mathrm{v}\mathit{erified}000000111101111111110 - \{\ \}.$$

shows that these sub-boxes were subsequently killed as well, and thus the entire sub-box $Z(000000111101111111)$ has been killed.

Instead of immediately working on killing the top-level box, we subdivide in the six co-ordinate directions to get the 64 sub-boxes

$$Z(000000), Z(000001), Z(000010), Z(000011), \ldots, Z(111111).$$

We then throw out the ones with fourth co-ordinate equal to 1 (see condition 'conjugate'), leaving the 32 sub-boxes

$$Z(000000), Z(000001), Z(000010), Z(000011), \ldots, Z(111011).$$

We then use *v*erify to kill these.

The choices in *v*erify are made for it by a sequence of integers given as input. The sequence of integers containing the directions for killing $Z(000000)$ is contained in the file data/000000 (actually, data/000000.d). In such a sequence, 0 tells *v*erify to subdivide the present box (by $x_i = c$), to position itself on the "left-hand" box ($x_i \leq c$) created by that subdivision, and to read in the next integer in the sequence. A positive integer n tells *v*erify to kill directly the sub-box it is positioned at, using the condition (and killerword, if necessary) on line n in the "conditionlist" file, and then to position itself at the "next" natural sub-box. Now, -1 tells *v*erify to omit the sub-box, and mark it as skipped (the sequence of integers used in killing the skipped box $Z(s)$ is contained in a file data/s).

The checker program *v*erify, its inputs, and the list of conditions are available from the *A*nnals of Mathematics web site.

Example 4.4. To illustrate the checking process in action, this is a (non-representative) example, which shows how the sub-box $Z(s)$ (minus a hole) is killed, where

$$s = 0010001100011101110011111000101111111011111 \\ 0011100111100000111101111011 1.$$

The input associated with this sub-box is

$$(0, 0, 0, 1929, 12304, 0, 0, 7, 0, 1965, 0, 1929, 1929, 1996, -1),$$

which causes the program to kill $Z(s)$ in the following fashion:

kill $Z(s)$:
 kill $Z(s0)$:
  kill $Z(s00)$:
   kill $Z(s000)$ with condition 1929 = "L(FwFWFWfWFWFwFwfww)"
   kill $Z(s001)$ with condition 12304 = "L(FwfWFFWFwFwfwfWfwfw)"
  kill $Z(s01)$:
   kill $Z(s010)$:
    kill $Z(s0100)$ with condition 7 = "L(w)"
    kill $Z(s0101)$:
     kill $Z(s01010)$ with condition 1965 = "L(fwFwFWFFWFwFwfwww)"
     kill $Z(s01011)$:
      kill $Z(s010110)$ with condition 1929
      kill $Z(s010111)$ with condition 1929

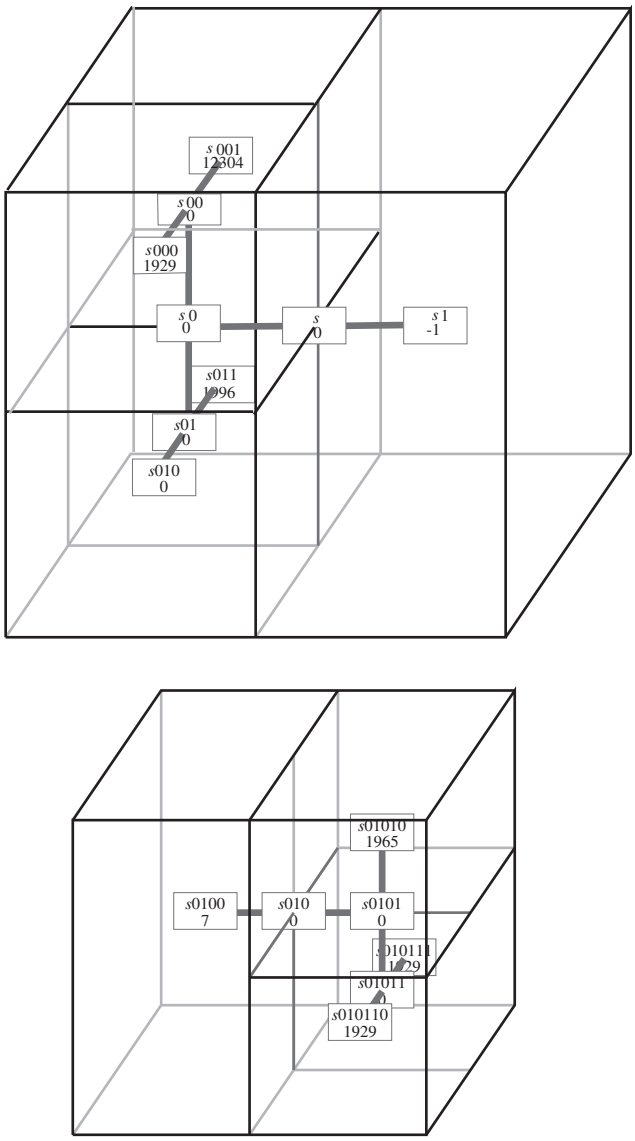


Figure 4.1: Six levels of subdivision, in two projections, with all the trimmings.

kill $Z(s011)$ with condition 1996 = "L(FwFwFWFWfWFWFwFwfww)"
omit $Z(s1)$
as shown in Figure 4.4.

$Z(s1)$ is ignored, so the checker would indicate this omission in its report. In fact, $Z(s1)$ is one of the 11 exceptional sub-boxes (seven boxes after joining abutters), specifically X_{5a}, hence killed automatically.

The use of condition "L(w)" so deep in the tree is unusual. In this case, it is because the

manifold in the exceptional sub-box has $length(f) = length(w)$, so that the program will frequently come to places where it can bound $length(f) > length(w)$ nearby.

The sequence $0, 1929, 1929$ in the input for $Z(s)$ tells the checker to subdivide $Z(s01011)$, and then use the condition 1929 on both halves to kill them separately, thereby killing $Z(s01011)$. As mentioned above, the reason for carrying out this subdivision is that the remainder/error bound in the calculation for $Z(s01011)$ using condition 1929 was not good enough to prove that the sub-box is killed directly.

In the input for $Z(s)$ we could have replaced $0, 1929, 1929$ with 1929 alone and then the checker program itself would be smart enough to carry out the subdivision after 1929 failed to kill the sub-box $Z(s01011)$. This recursive subdivision tool is quite useful when dealing with the remainder/error term—if a killerword barely misses killing off a sub-box, then recursively subdivide the sub-box and use the same killerword on the pieces until it succeeds. We note that the theoretical error arising from using a first-order Taylor approximation is likely to be significantly improved by subdivision, whereas the round-off error is relatively unaffected because the killerword used is unchanged (hence the number of mathematical operations performed is unchanged).

The binary numbers used by the computer require too much space to print. In the example calculation which follows, we instead use decimal representations (although we print fewer digits than could be gotten from the 53 binary digits used for the actual calculations).

The sub-box $Z(s01011)$ is the region where

$$\left(\begin{array}{l} -1.381589027741\ldots \le \mathrm{R}e(L') \le -1.379848991182\ldots \\ -1.378124546093\ldots \le \mathrm{R}e(D') \le -1.376574349753\ldots \\ \quad 0.999893182771\ldots \le \mathrm{R}e(R') \le 1.001274250703\ldots \\ -2.535837191243\ldots \le \mathrm{I}m(L') \le -2.534606799593\ldots \\ \quad 2.535404997792\ldots \le \mathrm{I}m(D') \le -2.534308843448\ldots \\ -0.001953125000\ldots \le \mathrm{I}m(R') \le 0.000000000000\ldots \end{array}\right)$$

At this point, we would like to compute

$$f,\ w,\ g = f^{-1}wf^{-1}w^{-1}f^{-1}w^{-1}fw^{-1}f^{-1}w^{-1}f^{-1}wf^{-1}wfww,\ length(g),$$

and so on. However, these items take on values over an entire sub-box and thus are computed via AffApproxes (first-order Taylor approximations with remainder/error bounds), which are not formally defined until the next section. We complete Example 4.4 at the end of Chapter 5.

Remark 4.5. . For those planning on looking at the program *v*erify we now tie in the above description of its workings to a portion of the actual code in the program. We note that the CWeb version of *v*erify is extensively documented, and is organized so that the most important details are presented first.

If the executable version of *v*erify is called *v*erify and we are in the correct place with respect to the location of the data, then a typical UNIX command line would be

zcat data/000000.gz — verify 000000 ¿ output000000

This would run *v*erify at the node 000000, and, when needed, would pipe in the unzipped data from data/000000.gz. This unzipped data contains the tree decomposition of the parameter space at the sub-box 000000. The output from *v*erify would be redirected to the file output000000.

In *v*erify, main would check for syntax errors in the command line, and if there were no such errors, would read the location 000000 into the character array where and compute that the depth of where was 6, which means that 000000 contains six subdivisions. It would then immediately print

verified 000000 - {

into the file output000000, and then call the function *v*erify, as follows:

verify(where, depth, 0);

The function verify(where, depth, autocode) is now invoked; this time with autocode equal to 0. *V*erify would first check that depth was not too deep. Next, *v*erify checks if autocode is equal to 0, which it is, so it reads in the next (in this case, the first) integer from the unzipped file data/000000.gz, and sets code equal to this integer. Now, *v*erify recursively calls itself on the left child (0000000) of the where box and the right child (0000001) of the where box:

```
where[depth] = '0';
verify(where, depth + 1, code);
where[depth] = '1';
verify(where, depth + 1, code);
```

In general, verify(where, depth, autocode) does the following. It checks to see that depth is not too deep. Then if autocode is equal to 0, it recursively calls itself on its left and right children. If autocode is not equal to 0, then code is set equal to autocode, and three possibilities can occur. Either,

1) code is less than zero, in which case we are at a sub-box to be skipped, and *v*erify prints out its location (where) in output000000 and recursively moves on to the next node in the tree, or

2) code is greater than zero and it invokes a condition/killerword from the file *c*onditionlist which kills the entire sub-box where in which case verify simply recursively moves on to the next node in the tree, or

3) code is greater than zero and it invokes a condition/killerword from the file *c*onditionlist which does not kill the entire sub-box where, in which case verify subdivides the sub-box

where and recursively calls itself on the left child and the right child, using the same code:

```
where[depth] = '0';
verify(where, depth + 1, code);
where[depth] ='1';
verify(where, depth + 1, code);
```

In this way *v*erify tests the entire starting box, in this case the sub-box 000000, and if successful at killing it minus the omissions which it prints out, it finishes main by printing out a right bracket into output000000.

Chapter 5

Affine approximations

Remark 5.1. To show that a sub-box of the parameter box $\mathcal{W}$ is killed by one of the interesting conditions (plus associated killerword) we need to show that at each point in the sub-box, the killerword evaluated at that point satisfies the given condition (see Chapter 4). That is, we are simply analyzing a certain function from the sub-box to $\mathbf{C}$.

As described in Remark 5.5, this analysis can be pulled back from the sub-box in question to the closed polydisc $A = \{(z_0, z_1, z_2) \in \mathbf{C}^3 : |z_k| \leq 1 \text{ for } k \in \{0, 1, 2\}\}$. Loosely, we will analyze such a function on A by using Taylor approximations consisting of an affine approximating function together with a bound on the "error" in the approximation (this could also be described as a "remainder bound"). This "error" is separate from round-off error, which will be analyzed in Chapters 6 and 7.

Problem 5.2. There are two immediate problems likely to arise from this Taylor approximation approach. The first problem is the appearance of unpleasant functions such as Arccosh. We have already taken care of this problem by "exponentiating" our preliminary parameter space $\mathcal{P}$. This resulted in all functions under consideration being built up from the co-ordinate functions L', D', and R' on $\mathcal{W}$ by means of the elementary operations $+, -, \times, /, \sqrt{\cdot}$.

Second, for a given "built-up function" the computer needs to be able to compute the Taylor approximation, and the error term. This will be handled by developing combination formulas for elementary operations (see the propositions below). Specifically, given two Taylor approximations with error terms representing functions g and h and an elementary operation on g and h, we will show how to get the Taylor approximation with error term for the resultant function from the two original Taylor approximations.

A similar approach was developed independently by Figuereido and Stolfi (see [FS]).

Remark 5.3. We set up the Taylor approximation approach rigorously as follows in Definition 5.4. The notation will be a bit unusual, but we are motivated by a desire to stay close to the notation used in the checker computer programs, *v*erify. However, it should be pointed out that the formulas in this chapter will be superseded by the ones in Chapter 7,

which incorporate a round-off error analysis. It is the Chapter 7 formulas that are used in *v*erify.

Definition 5.4. An *A*ffApprox x is a five-tuple $(x.f;\ x.f_0,\ x.f_1,\ x.f_2;\ x.e)$, consisting of four complex numbers $x.f,\ x.f_0,\ x.f_1,\ x.f_2$ and one real number $x.e$, which represents all functions $g : A \to \mathbf{C}$ such that

$$|g(z_0, z_1, z_2) - (x.f + x.f_0 z_0 + x.f_1 z_1 + x.f_2 z_2)| \leq x.e$$

for all $(z_0, z_1, z_2) \in A$. That is, x represents all functions from A to $\mathbf{C}$ that are $x.e$-well-approximated by the affine function $x.f + x.f_0 z_0 + x.f_1 z_1 + x.f_2 z_2$. We will denote this set of functions associated with x by $S(x)$.

Remark 5.5. As mentioned in Remark 5.1, given a sub-box to analyze, instead of working with functions defined on the sub-box, we will work with corresponding functions defined on A. Specifically, rather than build up a function by elementary operations performed on the co-ordinate functions L', D', R' restricted to the given sub-box, we will perform the elementary operations on the following functions defined on A,

$$(p_0 + ip_3; s_0 + is_3, 0, 0; 0)\ \ (p_1 + ip_4; 0, s_1 + is_4, 0; 0)\ \ (p_2 + ip_5; 0, 0, s_2 + is_5; 0)$$

where $(p_0{+}ip_3, p_1{+}ip_4, p_2{+}ip_5)$ is the center of the sub-box in question, and the s_i describe the six dimensions of the box. In the computer programs, these three functions are called *a*long, *o*rtho, and *w*hirle, respectively, and p_i and s_i are denoted *p*os[i] and *s*ize[i], respectively.

After the following remarks, we state and prove the combination formulas.

Remarks 5.6. i)i The reader may find useful the sequential correspondence between propositions of this Chapter and those of Chapter 7.

ii) The negation of a set of functions is the set consisting of the negatives of the original functions, and similarly for other operations.

iii) The propositions that follow include in their statements the definitions of the various operations on AffApproxes. What needs to be proved is that the S functions behave as expected. For example, we need to show that under the definition given for addition, the set of functions $S(x + y)$ contains all functions obtained by adding a function from $S(x)$ to a function from $S(y)$.

Proposition 5.7. *(unary minus) If x is an AffApprox, then $S(-x) = -(S(x))$ where*

$$-x \equiv (-x.f;\ -x.f_0,\ -x.f_1,\ -x.f_2;\ x.e).$$

Proof.

$$|g(z_0, z_1, z_2) - (x.f + x.f_0 z_0 + x.f_1 z_1 + x.f_2 z_2)| \leq e$$

if and only if

$$|-g(z_0, z_1, z_2) - (-x.f - x.f_0 z_0 - x.f_1 z_1 - x.f_2 z_2)| \leq e. \qquad \square$$

Proposition 5.8. *(addition)If x and y are AffApproxes, then $S(x+y) \supseteq S(x)+S(y)$, where*

$$x + y \equiv (x.f + y.f; x.f_0 + y.f_0, x.f_1 + y.f_1, x.f_2 + y.f_2; x.e + y.e).$$

Proof. If $g \in S(x)$ and $h \in S(y)$ then we must show that $g + h \in S(x + y)$.

$$\begin{aligned}
&|(g+h)(z_0,z_1,z_2)\\
&\quad -((x.f+y.f)+(x.f_0+y.f_0)z_0+(x.f_1+y.f_1)z_1+(x.f_2+y.f_2)z_2)|\\
&\qquad \le |g(z_0,z_1,z_2)-(x.f+(x.f_0)z_0+(x.f_1)z_1+(x.f_2)z_2)|\\
&\qquad\quad +|h(z_0,z_1,z_2)-(y.f+(y.f_0)z_0+(y.f_1)z_1+(y.f_2)z_2)|\\
&\qquad \le x.e+y.e.
\end{aligned}$$

□

Proposition 5.9. *(subtraction)If x and y are AffApproxes, then $S(x-y) \supseteq S(x)-S(y)$, where*

$$x - y \equiv (x.f - y.f; x.f_0 - y.f_0, x.f_1 - y.f_1, x.f_2 - y.f_2; x.e + y.e).$$

In what follows, "double" refers to a real number, and has an associated AffApprox, with the last four entries zero. When we do machine arithmetic in Chapters 6 and 7, doubles will be machine numbers.

Proposition 5.10. *(addition of an AffApprox and a double) If x is an AffApprox and y is a double, then $S(x+y) \supseteq S(x)+S(y)$, where*

$$x + y \equiv (x.f + y; x.f_0, x.f_1, x.f_2; x.e).$$

Proposition 5.11. *(subtraction of a double from an AffApprox) If x is an AffApprox and y is a double, then $S(x-y) \supseteq S(x)-S(y)$, where*

$$x - y \equiv (x.f - y; x.f_0, x.f_1, x.f_2; x.e).$$

Proposition 5.12. *(multiplication) If x and y are AffApproxes, then $S(x\times y) \supseteq S(x)\times S(y)$, where*

$$\begin{aligned}
x \times y \;\equiv\; & (x.f \times y.f; x.f \times y.f_0 + x.f_0 \times y.f,\\
& \quad x.f \times y.f_1 + x.f_1 \times y.f, x.f \times y.f_2 + x.f_2 \times y.f;\\
& \quad (size(x) + x.e) \times (size(y) + y.e) + (|x.f| \times y.e + x.e \times |y.f|))
\end{aligned}$$

with $size(x) = |x.f_0| + |x.f_1| + |x.f_2|$ and $size(y) = |y.f_0| + |y.f_1| + |y.f_2|$.

Proof. If $g \in S(x)$ and $h \in S(y)$ then we must show that $g \times h \in S(x \times y)$. That is, we need to show

$$\begin{aligned}
&|(g\times h)(z_0,z_1,z_2) - ((x.f\times y.f)+(x.f\times y.f_0 + x.f_0\times y.f)z_0\\
&\quad + (x.f\times y.f_1 + x.f_1\times y.f)z_1 + (x.f\times y.f_2 + x.f_2\times y.f)z_2)|\\
&\le (size(x)+x.e)\times(size(y)+y.e) + (|x.f|\times y.e + x.e\times|y.f|).
\end{aligned}$$

Note that for any point $(z_0, z_1, z_2) \in A$ and any functions $g \in S(x)$ and $h \in S(y)$ we can find complex numbers u, v with $|u| \leq 1$ and $|v| \leq 1$, such that

$$g(z_0, z_1, z_2) = x.f + (x.f_0 z_0 + x.f_1 z_1 + x.f_2 z_2) + (x.e)u$$

and

$$h(z_0, z_1, z_2) = y.f + (y.f_0 z_0 + y.f_1 z_1 + y.f_2 z_2) + (y.e)v.$$

Multiplying out, we see that

$$\begin{aligned}
&(g \times h)(z_0, z_1, z_2) \\
&\quad = \quad (x.f \times y.f) + (x.f \times y.f_0 + x.f_0 \times y.f)z_0 \\
&\qquad + (x.f \times y.f_1 + x.f_1 \times y.f)z_1 + (x.f \times y.f_2 + x.f_2 \times y.f)z_2 \\
&\qquad + (x.f \times y.e)v + (x.e \times y.f)u + ((x.f_0 z_0 + x.f_1 z_1 + x.f_2 z_2) + (x.e)u) \\
&\qquad \times ((y.f_0 z_0 + y.f_1 z_1 + y.f_2 z_2) + (y.e)v).
\end{aligned}$$

Hence,

$$\begin{aligned}
&|(g \times h)(z_0, z_1, z_2) - ((x.f \times y.f) \\
&\qquad + ((x.f \times y.f_0 + x.f_0 \times y.f)z_0 \\
&\qquad + (x.f \times y.f_1 + x.f_1 \times y.f)z_1 + (x.f \times y.f_2 + x.f_2 \times y.f)z_2))| \\
&\quad \leq \quad (|x.f|y.e + x.e|y.f|) + (\mathrm{size}(x) + x.e) \times (\mathrm{size}(y) + y.e).
\end{aligned}$$

□

Proposition 5.13. *(an AffApprox multiplied by a double) If x is an AffApprox and y is a double, then $S(x \times y) \supseteq S(x) \times S(y)$, where*

$$x \times y \equiv (x.f \times y; x.f_0 \times y, x.f_1 \times y, x.f_2 \times y; x.e \times |y|).$$

Proposition 5.14. *(division) If x and y are AffApproxes with $|y.f| > size(y) + y.e$, then $S(x/y) \supseteq S(x)/S(y)$, where*

$$\begin{aligned}
x/y \quad \equiv \quad & (x.f/y.f; (-x.f \times y.f_0 + x.f_0 \times y.f)/((y.f)^2), \\
& \qquad (-x.f \times y.f_1 + x.f_1 \times y.f)/((y.f)^2), \\
& \qquad (-x.f \times y.f_2 + x.f_2 \times y.f)/((y.f)^2); \\
& \qquad (|x.f| + size(x) + x.e)/(|y.f| - (size(y) + y.e)) \\
& - ((|x.f|/|y.f| + size(x)/|y.f|) + |x.f|size(y)/(|y.f||y.f|))).
\end{aligned}$$

Proof. For notational convenience, denote $(x.f_0 z_0 + x.f_1 z_1 + x.f_2 z_2)$ by $x.f_k z_k$ and similarly for $y.f_k z_k$ and so on. As above, note that for any point $(z_0, z_1, z_2) \in A$ and any functions $g \in S(x)$ and $h \in S(y)$ we can find complex numbers u, v with $|u| \leq 1$ and $|v| \leq 1$, such that

$$g(z_0, z_1, z_2) = x.f + (x.f_k z_k) + (x.e)u$$

and

$$h(z_0, z_1, z_2) = y.f + (y.f_k z_k) + (y.e)v.$$

We compare $(g/h)(z_0, z_1, z_2)$ with its putative affine approximation. That is, we analyze

$$\big|(x.f + (x.f_k z_k) + (x.e)u)/(y.f + (y.f_k z_k) + (y.e)v) \\ - ((x.f/y.f) + \frac{(x.f_k)y.f - x.f(y.f_k)}{(y.f)^2} z_k)\big|.$$

Putting this over a common denominator of $|((y.f)^2)(y.f + (y.f_k z_k) + (y.e)v)|$ and cancelling equal terms (in the numerator) we are left with a quotient whose numerator is

$$|x.e((y.f)^2)u - (x.f_k)y.f(y.f_k)z_k - x.f((y.f_k)^2)z_k \\ + (x.f)y.f(y.e)v + x.f_k(y.f)y.e(v)z_k - x.f(y.f_k)y.e(v)z_k|.$$

We must show this (first) quotient is bounded by

$$(|x.f| + size(x) + x.e)/(|y.f| - (size(y) + y.e)) \\ - ((|x.f|/|y.f| + size(x)/|y.f|) + |x.f|size(y)/(|y.f||y.f|)).$$

Putting this over a common denominator of $|(y.f)|^2(|y.f| - (size(y) + y.e))$ and cancelling equal terms (in the numerator) we are left with a second quotient, whose numerator is

$$x.e|y.f|^2 - (-|x.f||y.f|y.e - size(x)|y.f|(size(y) + y.e) - |x.f|size(y)(size(y) + y.e))$$

and we see that all terms in this numerator are positive. Further, the terms in the numerators of the first and second quotients correspond in a natural way, and each term in the numerator of the second quotient is greater than or equal to the absolute value of its corresponding term in the numerator of the first quotient.

Finally, because the denominator in the second quotient is less than or equal to the absolute value of the denominator in the first quotient, we see that the absolute value of the first quotient is less than or equal to the second quotient, as desired. □

Proposition 5.15. *(division of a double by an AffApprox) If x is a double and y is an AffApprox with $|y.f| > size(y) + y.e$, then $S(x/y) \supseteq S(x)/S(y)$, where*

$$x/y \equiv (x/y.f; -x \times y.f_0/((y.f)^2), -x.f \times y.f_1/((y.f)^2), -x.f \times y.f_2/((y.f)^2); \\ (|x|/(|y.f| - (size(y) + y.e)) - (|x|/|y.f| + |x|size(y)/(|y.f||y.f|))).$$

Proposition 5.16. *(division of an AffApprox by a double) If x is an AffApprox and y is a double with $|y| > 0$, then $S(x/y) \supseteq S(x)/S(y)$, where*

$$x/y \equiv (x.f/y; x.f_0/y, x.f_1/y, x.f_2/y; x.e/|y|).$$

Finally, we do the square root.

Proposition 5.17. *(square root) If x is an AffApprox with $|x.f| > size(x) + x.e$, then $S(\sqrt{x}) \supseteq \sqrt{S(x)}$, where*

$$\begin{aligned}\sqrt{x} \;=\; & \left(\sqrt{x.f}; \frac{x.f_0}{2\sqrt{x.f}}, \frac{x.f_1}{2\sqrt{x.f}}, \frac{x.f_2}{2\sqrt{x.f}}; \right. \\ & \left. \sqrt{|x.f|} - \left(\frac{size(x)}{2\sqrt{|x.f|}} + \sqrt{|x.f| - (size(x) + x.e)}\right)\right).\end{aligned}$$

If $|x.f| \leq \mathrm{s}ize(x) + x.e$ then we use the crude estimate

$$\left(0; 0, 0, 0; \sqrt{|x.f| + \mathrm{s}ize(x) + x.e}\right).$$

The branch of the square root of a complex number is determined by the construction of the square root of a complex in Proposition 6.23. In fact, the square root is in the first or fourth quadrant.

Proof. As above, note that for any point $(z_0, z_1, z_2) \in A$ and any function $g \in S(x)$ we can find a complex number u with $|u| \leq 1$, such that

$$g(z_0, z_1, z_2) = x.f + (x.f_k z_k) + (x.e)u.$$

Also, because $|x.f| > \mathrm{s}ize(x) + x.e.$ we see that the argument of $x.f + (x.f_k z_k) + (x.e)u$ is within $\pi/2$ of the argument of $x.f$, and therefore, we can require that $\sqrt{g(z_0, z_1, z_2)}$ have argument within $\pi/4$ of the argument of $\sqrt{x.f}$.

We need to show that

$$\begin{aligned}&\left|\sqrt{x.f + x.f_k z_k + (x.e)u} - (\sqrt{x.f} + \frac{x.f_k z_k}{2\sqrt{x.f}})\right| \\ &\qquad \leq \sqrt{|x.f|} - \left(\frac{\mathrm{s}ize(x)}{2\sqrt{|x.f|}} + \sqrt{|x.f| - (\mathrm{s}ize(x) + x.e)}\;\right).\end{aligned}$$

Or, after we multiply both sides by $\sqrt{|x.f|}$,

$$\begin{aligned}&\left|\sqrt{x.f(x.f + x.f_k z_k + (x.e)u)} - (x.f + (x.f_k)z_k/2)\right| \\ &\qquad \leq (|x.f| - \mathrm{s}ize(x)/2) - \sqrt{|x.f|(|x.f| - (\mathrm{s}ize(x) + x.e))}.\end{aligned}$$

The two sides of the inequality are of the form $A - B$ and $C - D$, and we "simplify" by multiplying by $\frac{A+B}{A+B}$ and $\frac{C+D}{C+D}$. We now show that the (absolute value of the) left-hand numerator is less than or equal to the right-hand numerator. Later, we will show that

the (absolute value of the) left-hand denominator is larger than or equal to the right-hand denominator. The left-hand numerator is

$$\begin{aligned}
&|x.f(x.f + x.f_k z_k + (x.e)u) - (x.f + (x.f_k)z_k/2)^2| \\
&\qquad = |(x.f)^2 + x.f(x.f_k)z_k + x.f(x.e)u - (x.f)^2 \\
&\qquad\quad -x.f(x.f_k)z_k - ((x.f_k)^2)(z_k)^2/4| \\
&\qquad = |x.f(x.e)u - ((x.f_k)^2)(z_k)^2/4|.
\end{aligned}$$

The right-hand numerator is

$$\begin{aligned}
&(|x.f| - size(x)/2)^2 - |x.f|(|x.f| - (size(x) + x.e)) \\
&\quad = |x.f|^2 - |x.f|size(x) + size(x)^2/4 - |x.f|^2 + |x.f|size(x) + |x.f|x.e \\
&\quad = |x.f|x.e + size(x)^2/4.
\end{aligned}$$

So the left-hand numerator is indeed less than or equal to the right-hand numerator.

We now compare the denominators, but only after dividing each by $\sqrt{|x.f|}$. The left-hand denominator is

$$\left|\sqrt{x.f + x.f_k z_k + (x.e)u} + \left(\sqrt{x.f} + \frac{x.f_k z_k}{2\sqrt{x.f}}\right)\right|$$

while the right-hand denominator is

$$\sqrt{|x.f|} - \frac{size(x)}{2\sqrt{|x.f|}} + \sqrt{|x.f| - (size(x) + x.e)}.$$

The claim that the left-hand denominator is greater than or equal to the right-hand denominator is a bit complicated. First, compare the $\sqrt{x.f}$ term and the $\sqrt{|x.f|}$ terms. They are the same distance from the origin. Next, note that as z_k and u take on all relevant values, $x.f + x.f_k z_k + (x.e)u$ describes a disk centered at $x.f$ with radius less than $|\sqrt{x.f}|$. Hence, $\sqrt{x.f + x.f_k z_k + (x.e)u}$ describes a convex set containing $\sqrt{x.f}$. This set is symmetric about the line joining the origin and $\sqrt{x.f}$. Further, $\sqrt{x.f} + \sqrt{x.f + x.f_k z_k + (x.e)u}$ describes a convex set containing $2\sqrt{x.f}$. This set is also symmetric about the line joining the origin and $\sqrt{x.f}$. It is easy enough to see that no points on this convex symmetric set get closer to the origin than $\sqrt{|x.f|} + \sqrt{|x.f| - (size(x) + x.e)}$.

Finally, because $|\frac{x.f_k z_k}{2\sqrt{x.f}}| \leq \frac{size(x)}{2\sqrt{|x.f|}}$, no points of

$$\sqrt{x.f} + \sqrt{x.f + x.f_k z_k + (x.e)u} + \frac{x.f_k z_k}{2\sqrt{x.f}}$$

can get closer to the origin than

$$\sqrt{|x.f|} + \sqrt{|x.f| - (size(x) + x.e)} - \frac{size(x)}{2\sqrt{|x.f|}}.$$

□

Example 5.18. (Continuation of Example 4.4). We can now complete the analysis begun in Example 4.4, because we can describe f and w as 2-by-2 matrices of AffApproxes. We note the minor quibble that the full definition of AffApprox is given in Chapter 7, where round-off error is incorporated into the remainder/error-bound term.

For convenience, we repeat the description of the sub-box under investigation. The sub-box $Z(s01011)$ with

$$s = 00100011000111011100111100010111111110111110011100111100000111101111 0111$$

is the region where

$$\begin{pmatrix} -1.381589027741\ldots \leq \mathrm{R}e(L') \leq -1.379848991182\ldots \\ -1.378124546093\ldots \leq \mathrm{R}e(D') \leq -1.376574349753\ldots \\ 0.999893182771\ldots \leq \mathrm{R}e(R') \leq 1.001274250703\ldots \\ -2.535837191243\ldots \leq \mathrm{I}m(L') \leq -2.534606799593\ldots \\ 2.535404997792\ldots \leq \mathrm{I}m(D') \leq -2.534308843448\ldots \\ -0.001953125000\ldots \leq \mathrm{I}m(R') \leq 0.000000000000\ldots \end{pmatrix}.$$

For this sub-box, we get (printing only 10 decimal places, for visual convenience):

$$f = \begin{bmatrix} \begin{pmatrix} -0.8677851121 + i1.4607429651; \\ 0.0000248810 - i0.0003125810, \\ 0.0000000000 + i0.0000000000, \\ 0.0000000000 + i0.0000000000; \\ 0.0000000289 \end{pmatrix} & \begin{pmatrix} 0.0000000000 + i0.0000000000; \\ 0.0000000000 + i0.0000000000, \\ 0.0000000000 + i0.0000000000, \\ 0.0000000000 + i0.0000000000; \\ 0.0000000000 \end{pmatrix} \\ \begin{pmatrix} 0.0000000000 + i0.0000000000; \\ 0.0000000000 + i0.0000000000, \\ 0.0000000000 + i0.0000000000, \\ 0.0000000000 + i0.0000000000; \\ 0.0000000000 \end{pmatrix} & \begin{pmatrix} -0.3006023265 - i0.5060039953; \\ -0.0000909686 - i0.0000593570, \\ 0.0000000000 + i0.0000000000, \\ 0.0000000000 + i0.0000000000; \\ 0.0000000301 \end{pmatrix} \end{bmatrix}$$

and

$$w = \left[\begin{pmatrix} -0.5845111829 + i0.4773282853; \\ 0.0000000000 + i0.0000000000, \\ -0.0000296707 - i0.0001657332, \\ -0.0004345111 - i0.0001209539; \\ 0.0000002590 \\ -0.2832291572 + i0.9833572297; \\ 0.0000000000 + i0.0000000000, \\ 0.0000515806 - i0.0001129408, \\ -0.0005778031 + i0.0002005440; \\ 0.0000002806 \end{pmatrix} \begin{pmatrix} -0.2840228472 + i0.9825063583; \\ 0.0000000000 + i0.0000000000, \\ 0.0000516606 - i0.0001128245, \\ 0.0005776611 - i0.0001998632; \\ 0.0000006462 \\ -0.5846352333 + i0.4764792236; \\ 0.0000000000 + i0.0000000000, \\ -0.0000294917 - i0.0001656653, \\ 0.0004341392 + i0.0001213070; \\ 0.0000005286 \end{pmatrix} \right].$$

Calculation of $g = f^{-1}wf^{-1}w^{-1}f^{-1}w^{-1}fw^{-1}f^{-1}w^{-1}f^{-1}wf^{-1}wfww$ gives

$$g = \left[\begin{pmatrix} -0.5764337542 + i0.4752708071; \\ -0.0031657223 - i0.0001436786, \\ -0.0017723577 + i0.0000352928, \\ -0.0011623491 + i0.0017516088; \\ 0.0008229225 \\ -0.2861207992 + i0.9766064999; \\ -0.0002777968 + i0.0020330488, \\ 0.0000837571 + i0.0010241875, \\ 0.0028322367 - i0.0005972336; \\ 0.0018172437 \end{pmatrix} \begin{pmatrix} -0.2704033973 + i0.9822741250; \\ -0.0045902952 - i0.0019135041, \\ -0.0026219461 - i0.0007506230, \\ -0.0002823450 + i0.0033805602; \\ 0.0008037640 \\ -0.5861133046 + i0.4624368851; \\ -0.0021932627 + i0.0040523411, \\ -0.0008612361 + i0.0022394639, \\ 0.0061581377 - i0.0005862070; \\ 0.0017738513 \end{pmatrix} \right].$$

We then get

$$\mathrm{l}ength(g) = \begin{pmatrix} -1.3588762105 - i2.4897230182; \\ 0.0030210500 - i0.0182284729, \\ 0.0007938572 - i0.0096614614, \\ -0.0122034521 + i0.0074353043; \\ 0.0080071969 \end{pmatrix}$$

and

$$\frac{\mathrm{l}ength(g)}{L'} = \begin{pmatrix} 0.9825397896 - i0.0008933519; \\ 0.0053701602 + i0.0037789019, \\ 0.0028076072 + i0.0018421952, \\ -0.0002400615 - i0.0049443045; \\ 0.0027802966 \end{pmatrix}.$$

This is not quite good enough to kill the sub-box, since $|length(g)/L'|$ can be as high as 1.0001951323.

When we subdivide $Z(s01011)$, we have to analyze two sub-boxes, $Z(s010110)$ and $Z(s010111)$. For $Z(s010110)$, the same calculation on the region

$$\begin{aligned}
-1.381589027741073400 \leq \mathrm{R}e(L') &\leq -1.379848991182205200 \\
-1.378124546093485700 \leq \mathrm{R}e(D') &\leq -1.376574349753672900 \\
0.999893182771602220 \leq \mathrm{R}e(R') &\leq 1.001274250703607400 \\
-2.535837191243490300 \leq \mathrm{I}m(L') &\leq -2.534606799593201600 \\
-2.535404997792558600 \leq \mathrm{I}m(D') &\leq -2.534308843448505900 \\
-0.001953125000000000 \leq \mathrm{I}m(R') &\leq -0.000976562500000000
\end{aligned}$$

gives

$$\frac{length(g)}{L'} = \begin{pmatrix} 0.9814518667 + i0.0008103446; \\ 0.0053616729 + i0.0037834001, \\ 0.0028027236 + i0.0018435245, \\ -0.0013175066 - i0.0032448794; \\ 0.0019033926 \end{pmatrix},$$

and we can then bound $\left|\frac{length(g)}{L'}\right| \leq 0.9967745579$, which kills $Z(s010110)$.

On $Z(s010111)$, the calculation gives

$$\frac{length(g)}{L'} = \begin{pmatrix} 0.9836225919 - i0.0025990177; \\ 0.0053786346 + i0.0037743930, \\ 0.0028124892 + i0.0018408583, \\ -0.0013333182 - i0.0032343347; \\ 0.0019044429 \end{pmatrix}$$

and $\left|\frac{length(g)}{L'}\right| \leq 0.9989610507$, which kills $Z(s010111)$.

Chapter 6

Complex numbers with round-off error

Remark 6.1. The theoretical method for proving Theorem 1.2 has been implemented via the computer program *v*erify, which is available, together with the relevant data sets, at the *A*nnals web site. To make this computer-aided proof rigorous, we needed to deal with round-off error in calculations.

One approach to round-off error would be to use interval arithmetic packages to carry out all calculations with floating-point machine numbers, or to generate our own version of these packages. However, it appears that this approach would be much too slow given the size of our collection of sub-boxes and conditions/killerwords.

To solve this problem of speed, we implement round-off error at a higher level of programing. That is, we incorporate round-off error directly into AffApproxes, which makes our error calculations more accurate, thereby avoiding much subdivision of sub-boxes. This necessitates that we incorporate round-off error directly into complex numbers as well. In this chapter we show how to do standard operations on complex numbers while keeping track of round-off error. In the next chapter we work with AffApproxes.

Definition 6.2. There are two types of complex numbers to consider:

1) An *X*Complex $x = (x.\mathrm{re}, x.\mathrm{i}m)$ corresponds to a complex number that is represented exactly; it simply consists of a real part and an imaginary part.

2) An *A*Complex $x = (x.\mathrm{re}, x.\mathrm{i}m; x.e)$ corresponds to an "interval" that contains the complex number in question. Thus, it consists of an XComplex and a floating-point number representing the error. In particular, the AComplex x represents the set $S(x)$ of complex numbers $\{w \,:\, |w-(x.\mathrm{re}+i(x.\mathrm{i}m))| \le x.e\}$. Note that $S(x)$ is also defined for an XComplex if we conceptualize an XComplex as an AComplex with $x.e = 0$.

Remark 6.3. In general, our operations act on XComplexes and produce AComplexes, or they act on AComplexes and produce AComplexes. In one case, the unary minus, an

XComplex goes to an XComplex. In the calculations that follow the effect on the error is the whole point.

Conventions 6.4. We begin, by writing down our basic rules, which follow easily from the IEEE-754 standard for machine arithmetic (see [IEEE]). (Actually, the "hypot" function $h(a,b)$, which computes by elaborate chicanery $\sqrt{a^2+b^2}$, is not part of the IEEE-754 standard, but satisfies the appropriate standard according to the documentation provided (see [K1]).) The operations here are on double-precision floating-point real numbers ("doubles") and we denote a true operation by the usual symbol and the associated machine operation by the same symbol in a circle, with two exceptions: a machine square root $\sqrt{a}$ is denoted $\sqrt[\circ]{a}$ and the machine version of the hypot function is denoted $h_\circ$. Perhaps a third exception is our occasional notation of true multiplication by the absence of a symbol.

There is a finite set of numbers (sometimes called "machine numbers") which are representable on the computer. With technicalities ignored, a nonzero floating-point number is represented by a fixed number of bits of which the first determines the sign of the number, the next m represent the exponent, and the remaining n represent the mantissa of the number. Because our nonzero numbers start with a 1, that means the n mantissa bits actually represent the next n binary digits after the 1. That is, the mantissa is actually $1.b_1b_2b_3...b_n$. The IEEE-754 standard calls for 64-bit doubles with $m = 11$ and $n = 52$. We define EPS to be 2^{-n}, in which case $EPS/2$ is $2^{-(n+1)}$.

The IEEE-754 standard states that the result of an operation is always the closest representable number to the true solution (as long as we are in the bounds of representable numbers). For example, for machine numbers a and b, we have $a \oplus b = m(a+b)$ where m is the function which takes the machine value of its argument (when it lies in the range of representable numbers). Thus, properties of the type

$$|(a+b)-(a\oplus b)| \le (EPS/2)|a+b|$$

follow immediately from the IEEE-754 standard, as long as we do not *u*nderflow or *o*verflow outside of the range of representable numbers. Specifically, underflow occurs when the result of an operation is smaller in absolute value than 2^{-1022}, and overflow occurs when the result of an operation is larger in absolute value than roughly 2^{1024} (see [IEEE], §7).

We further note that the formula

$$|(a+b)-(a\oplus b)| \le (EPS/2)|a\oplus b|$$

follows because the true answer has "exponent" which is less than or equal to the exponent of the machine answer. We reiterate, that in both cases, a and b are assumed to be machine numbers.

Of course, a machine operation such as $\oplus$ must act on doubles, while a "true" operation such as $+$ can act on reals (which includes doubles). In this chapter, long strings of inequalities will be used to prove the various propositions, and care was taken to ensure that machine operations act on machine numbers. In particular, the various variables appearing in the propositions are assumed to be doubles. The IEEE-754 standard provides

for conversions from decimal to binary (within the appropriate range, conversion is to the nearest representable number) and from binary to decimal. However, these are rarely used in this paper, although a trivial class of exceptions is provided by the decimal numbers in the conditions of Chapter 4

When calculations underflow or overflow outside of the range of representable numbers, we require that the computer inform us if either exception has occurred.

Basic Properties 6.5. (assuming no underflow and no overflow)

In the formulas that follow, a,b, and A are machine numbers and $1+k\times EPS = 1\oplus(k\otimes EPS)$ when k is an integer which is not huge in absolute value (that is, smaller than roughly 2^{50}). Thus, within the appropriate range, $1 + k \times EPS$ is a machine number. Similarly, $2^k \times A = 2^k \otimes A$ when k is an integer and $2^k \otimes A$ neither underflows nor overflows.

$$\begin{aligned}
|(a+b)-(a\oplus b)| &\le (EPS/2)|a+b|,\\
|(a+b)-(a\oplus b)| &\le (EPS/2)|a\oplus b|,\\
|(a-b)-(a\ominus b)| &\le (EPS/2)|a-b|,\\
|(a-b)-(a\ominus b)| &\le (EPS/2)|a\ominus b|,\\
|(a\times b)-(a\otimes b)| &\le (EPS/2)|a\times b|,\\
|(a\times b)-(a\otimes b)| &\le (EPS/2)|a\otimes b|,\\
|(a/b)-(a\oslash b)| &\le (EPS/2)|a/b|,\\
|(a/b)-(a\oslash b)| &\le (EPS/2)|a\oslash b|,\\
|\sqrt{a}-\mathring{\sqrt}{a}| &\le (EPS/2)|\sqrt{a}|,\\
|\sqrt{a}-\mathring{\sqrt}{a}| &\le (EPS/2)|\mathring{\sqrt}{a}|,\\
|h(a,b)-h_\circ(a,b)| &\le (EPS)|h(a,b)|,\\
|h(a,b)-h_\circ(a,b)| &\le (EPS)|h_\circ(a,b)|.
\end{aligned}$$

From these formulas, we immediately compute the following.

$$\begin{aligned}
(1-EPS/2)|a+b| &\le |a\oplus b| \le (1+EPS/2)|a+b|,\\
(1-EPS/2)|a\oplus b| &\le |a+b| \le (1+EPS/2)|a\oplus b|,\\
&\vdots\\
(1-EPS/2)|\sqrt{a}| &\le |\mathring{\sqrt}{a}| \le (1+EPS/2)|\sqrt{a}|,\\
(1-EPS/2)|\mathring{\sqrt}{a}| &\le |\sqrt{a}| \le (1+EPS/2)|\mathring{\sqrt}{a}|,\\
(1-EPS)|h(a,b)| &\le |h_\circ(a,b)| \le (1+EPS)|h(a,b)|,\\
(1-EPS)|h_\circ(a,b)| &\le |h(a,b)| \le (1+EPS)|h_\circ(a,b)|.
\end{aligned}$$

Of course, we can also get the following type of formula, which is sometimes convenient, for example, in the proof of Lemma 6.20:

$$\left(\frac{1}{1+\frac{EPS}{2}}\right)|a\oplus b| \le |a+b| \le \left(\frac{1}{1-\frac{EPS}{2}}\right)|a\oplus b|.$$

Before stating our propositions, we prove two lemmas.

Lemma 6.6. *(assuming no underflow and no overflow) For machine numbers a and b,*

$$(1-EPS)\otimes|a\oplus b| \le |a+b| \le (1+EPS)\otimes|a\oplus b|.$$

Analogous formulas hold for $-$, $*$, $/$, $\sqrt{}$.

Proof. Assume $a+b>0$. If $(1+EPS)\otimes(a\oplus b) < (a+b)$ then the machine number $(1+EPS)\otimes(a\oplus b)$ is a better approximation to $a+b$ than $a\oplus b$, because $(a\oplus b) < (1+EPS)\otimes(a\oplus b)$. This contradicts the IEEE standard. The case $a+b<0$ can be handled similarly, and the case $a+b=0$ is trivial, similarly for the left-hand inequality. □

Lemma 6.7.

$$(1+EPS/2)^a A \le (1+kEPS)\otimes A$$

where A is a nonnegative machine number, and a is a (not huge) integer, such that for a even, $k=\frac{a}{2}+1$ and for a odd, $k=\frac{a+1}{2}+1$.

Proof.

$$(1+EPS/2)^a A \le (1-EPS/2)(1+kEPS)A \le (1+kEPS)\otimes A.$$

The first inequality holds if a and k are as in the lemma, and the second inequality is a consequence of one of the formulas preceding Lemma 6.6 ($A\ge 0$). □

We now begin our construction of complex arithmetic. We will give proofs for most of the operations; the others should be straightforward to derive, or can be found in the *A*nnals web site.

Remarks 6.8. i) We remind the reader that all machine operations are on machine numbers, and that the various variables appearing in the propositions are assumed to be doubles.

ii) The propositions that follow include in their statements the definitions of the various operations (see Remark 5.6 iii).

Proposition 6.9. $(-X)$ *If x is an XComplex, then*

$$-x \equiv (-x.\mathrm{re}, -x.\mathrm{im}).$$

Proposition 6.10. $(X+D)$ *If x is an XComplex and d is a double, then $S(x+d)\supseteq S(x)+S(d)$, where*

$$x+d \equiv (x.\mathrm{re}\oplus d, x.\mathrm{im}; (EPS/2)\otimes|x.\mathrm{re}\oplus d|).$$

Proof. The error is bounded by

$$|(x.\mathrm{re} + d) - (x.\mathrm{re} \oplus d)| \leq (EPS/2)|x.\mathrm{re} \oplus d| = (EPS/2) \otimes |x.\mathrm{re} \oplus d|. \qquad \square$$

Proposition 6.11. $(X - D)$ *If x is an XComplex and d is a double, then $S(x - d) \supseteq S(x) - S(d)$, where*

$$x - d \equiv (x.\mathrm{re} \ominus d, x.\mathrm{im}; (EPS/2) \otimes |x.\mathrm{re} \ominus d|).$$

Proposition 6.12. $(X + X)$ *If x and y are XComplexes, then $S(x + y) \supseteq S(x) + S(y)$, where*

$$\begin{aligned} x + y \;\equiv\; & (x.\mathrm{re} \oplus y.\mathrm{re}, x.\mathrm{im} \oplus y.\mathrm{im}; (EPS/2) \\ & \otimes ((1 + EPS) \otimes (|x.\mathrm{re} \oplus y.\mathrm{re}| \oplus |x.\mathrm{im} \oplus y.\mathrm{im}|))). \end{aligned}$$

Proof. The error is bounded by

$$\begin{aligned} & |(x.\mathrm{re} + y.\mathrm{re}) - (x.\mathrm{re} \oplus y.\mathrm{re})| + |(x.\mathrm{im} + y.\mathrm{im}) - (x.\mathrm{im} \oplus y.\mathrm{im})| \\ & \quad \leq (EPS/2)(|x.\mathrm{re} \oplus y.\mathrm{re}| + |x.\mathrm{im} \oplus y.\mathrm{im}|) \\ & \quad \leq (EPS/2)((1 + EPS) \otimes (|x.\mathrm{re} \oplus y.\mathrm{re}| \oplus |x.\mathrm{im} \oplus y.\mathrm{im}|)) \\ & \quad = (EPS/2) \otimes ((1 + EPS) \otimes (|x.\mathrm{re} \oplus y.\mathrm{re}| \oplus |x.\mathrm{im} \oplus y.\mathrm{im}|)). \end{aligned}$$

To go from line 2 to line 3 we used Lemma 6.6. $\square$

Proposition 6.13. $(X - X)$ *If x and y are XComplexes, then $S(x - y) \supseteq S(x) - S(y)$, where*

$$\begin{aligned} x - y \;\equiv\; & (x.\mathrm{re} \ominus y.\mathrm{re}, x.\mathrm{im} \ominus y.\mathrm{im}; \\ & (EPS/2) \otimes ((1 + EPS) \otimes (|x.\mathrm{re} \ominus y.\mathrm{re}| \oplus |x.\mathrm{im} \ominus y.\mathrm{im}|))). \end{aligned}$$

Proposition 6.14. $(A + A)$ *If x and y are AComplexes, then $S(x + y) \supseteq S(x) + S(y)$, where*

$$\begin{aligned} x + y \;&\equiv\; (\mathrm{re}, \mathrm{im}; e) \text{ with} \\ \mathrm{re} \;&=\; x.\mathrm{re} \oplus y.\mathrm{re} \\ \mathrm{im} \;&=\; x.\mathrm{im} \oplus y.\mathrm{im} \\ e \;&=\; (1 + 2EPS) \otimes (((EPS/2) \otimes (|\mathrm{re}| \oplus |\mathrm{im}|)) \oplus (x.e \oplus y.e)). \end{aligned}$$

Proof. The error is bounded by the sum of the contributions from the real part, the imaginary part, and the two individual errors:

$$
\begin{aligned}
&|(x.\mathrm{re} \oplus y.\mathrm{re}) - (x.\mathrm{re} + y.\mathrm{re})| + |(x.\mathrm{im} \oplus y.\mathrm{im}) - (x.\mathrm{im} + y.\mathrm{im})| + (x.e + y.e).\\
&\quad \le (EPS/2)|x.\mathrm{re} \oplus y.\mathrm{re}| + (EPS/2)|x.\mathrm{im} \oplus y.\mathrm{im}| + (1 + EPS/2)(x.e \oplus y.e)\\
&\quad \le (1 + EPS/2)(EPS/2)(|x.\mathrm{re} \oplus y.\mathrm{re}| \oplus |x.\mathrm{im} \oplus y.\mathrm{im}|)\\
&\qquad\quad +(1 + EPS/2)(x.e \oplus y.e)\\
&\quad = (1 + EPS/2)((EPS/2)(|x.\mathrm{re} \oplus y.\mathrm{re}| \oplus |x.\mathrm{im} \oplus y.\mathrm{im}|) + (x.e \oplus y.e))\\
&\quad \le (1 + EPS/2)^2(((EPS/2)(|x.\mathrm{re} \oplus y.\mathrm{re}| \oplus |x.\mathrm{im} \oplus y.\mathrm{im}|)) \oplus (x.e \oplus y.e))\\
&\quad \le (1 + 2EPS) \otimes (((EPS/2) \otimes (|x.\mathrm{re} \oplus y.\mathrm{re}| \oplus |x.\mathrm{im} \oplus y.\mathrm{im}|)) \oplus (x.e \oplus y.e)).
\end{aligned}
$$

□

The precedence for machine operations is the same as that for true operations, so some parentheses are unnecessary and will often be omitted in what follows.

Proposition 6.15. $(A-A)$ *If x and y are AComplexes, then $S(x-y) \supseteq S(x) - S(y)$, where*

$$
\begin{aligned}
x - y &\equiv (\mathrm{re}, \mathrm{im}; e) \text{ with}\\
\mathrm{re} &= x.\mathrm{re} \ominus y.\mathrm{re}\\
\mathrm{im} &= x.\mathrm{im} \ominus y.\mathrm{im}\\
e &= (1 + 2EPS) \otimes (((EPS/2) \otimes (|\mathrm{re}| \oplus |\mathrm{im}|)) \oplus (x.e \oplus y.e)).
\end{aligned}
$$

Proposition 6.16. $(X \times D)$ *If x is an XComplex and d is a double, then $S(x \times d) \supseteq S(x) \times S(d)$, where*

$$
\begin{aligned}
x \times d &\equiv (\mathrm{re}, \mathrm{im}; e) \text{ with}\\
\mathrm{re} &= x.\mathrm{re} \otimes d\\
\mathrm{im} &= x.\mathrm{im} \otimes d\\
e &= (EPS/2) \otimes ((1 + EPS) \otimes (|\mathrm{re}| \oplus |\mathrm{im}|)).
\end{aligned}
$$

Proposition 6.17. (X/D) *If x is an XComplex and d is a double, then $S(x/d) \supseteq S(x)/S(d)$, where*

$$
\begin{aligned}
x/d &\equiv (\mathrm{re}, \mathrm{im}; e) \text{ with}\\
\mathrm{re} &= x.\mathrm{re} \oslash d\\
\mathrm{im} &= x.\mathrm{im} \oslash d\\
e &= (EPS/2) \otimes ((1 + EPS) \otimes (|\mathrm{re}| \oplus |\mathrm{im}|)).
\end{aligned}
$$

Proposition 6.18. $(X \times X)$ *If x and y are XComplexes, then $S(x \times y) \supseteq S(x) \times S(y)$, where*

$$\begin{aligned} x \times y &\equiv (\mathrm{re}, \mathrm{im}; e) \text{ with} \\ \mathrm{re} &= \mathrm{re}1 \ominus \mathrm{re}2, \text{ with } \mathrm{re}1 = x.\mathrm{re} \otimes y.\mathrm{re} \text{ and } \mathrm{re}2 = x.\mathrm{im} \otimes y.\mathrm{im} \\ \mathrm{im} &= \mathrm{im}1 \oplus \mathrm{im}2, \text{ with } \mathrm{im}1 = x.\mathrm{re} \otimes y.\mathrm{im} \text{ and } \mathrm{im}2 = x.\mathrm{im} \otimes y.\mathrm{re} \\ e &= EPS \otimes ((1 + 2EPS) \otimes ((|\mathrm{re}1| \oplus |\mathrm{re}2|) \oplus (|\mathrm{im}1| \oplus |\mathrm{im}2|))). \end{aligned}$$

Proof. The error is bounded by the sum of the contributions from the real part and the imaginary part:

$$\begin{aligned} &|(x.\mathrm{re} \times y.\mathrm{re} - x.\mathrm{im} \times y.\mathrm{im}) - ((x.\mathrm{re} \otimes y.\mathrm{re}) \ominus (x.\mathrm{im} \otimes y.\mathrm{im}))| \\ &+|(x.\mathrm{re} \times y.\mathrm{im} + x.\mathrm{im} \times y.\mathrm{re}) - ((x.\mathrm{re} \otimes y.\mathrm{re}) \oplus (x.\mathrm{im} \otimes y.\mathrm{im}))|. \end{aligned}$$

We want to bound this by a machine formula. Let us begin by bounding

$$|(x.\mathrm{re} \times y.\mathrm{re} - x.\mathrm{im} \times y.\mathrm{im}) - ((x.\mathrm{re} \otimes y.\mathrm{re}) \ominus (x.\mathrm{im} \otimes y.\mathrm{im}))|$$

by a machine formula:

$$\begin{aligned} &|(x.\mathrm{re} \times y.\mathrm{re} - x.\mathrm{im} \times y.\mathrm{im}) - ((x.\mathrm{re} \otimes y.\mathrm{re}) \ominus (x.\mathrm{im} \otimes y.\mathrm{im}))| \\ \leq\ &|((x.\mathrm{re} \times y.\mathrm{re}) - (x.\mathrm{im} \times y.\mathrm{im})) - ((x.\mathrm{re} \otimes y.\mathrm{re}) - (x.\mathrm{im} \otimes y.\mathrm{im}))| \\ &+|((x.\mathrm{re} \otimes y.\mathrm{re}) - (x.\mathrm{im} \otimes y.\mathrm{im})) - ((x.\mathrm{re} \otimes y.\mathrm{re}) \ominus (x.\mathrm{im} \otimes y.\mathrm{im}))| \\ \leq\ &|(x.\mathrm{re} \times y.\mathrm{re}) - (x.\mathrm{re} \otimes y.\mathrm{re})| + |(x.\mathrm{im} \times y.\mathrm{im}) - (x.\mathrm{im} \otimes y.\mathrm{im})| \\ &+(EPS/2)|(x.\mathrm{re} \otimes y.\mathrm{re}) - (x.\mathrm{im} \otimes y.\mathrm{im})| \\ \leq\ &(EPS/2)|(x.\mathrm{re} \otimes y.\mathrm{re})| + (EPS/2)|(x.\mathrm{im} \otimes y.\mathrm{im})| \\ &+(EPS/2)(|x.\mathrm{re} \otimes y.\mathrm{re}| + |x.\mathrm{im} \otimes y.\mathrm{im}|) \\ =\ &(EPS/2)(2)(|x.\mathrm{re} \otimes y.\mathrm{re}| + |x.\mathrm{im} \otimes y.\mathrm{im}|) \\ \leq\ &EPS(1 + EPS/2)(|x.\mathrm{re} \otimes y.\mathrm{re}| \oplus |(x.\mathrm{im} \otimes y.\mathrm{im}|). \end{aligned}$$

Almost the exact same calculation produces the analogous formula for the imaginary contribution, and we now combine the two to get a bound on the total error.

$$\begin{aligned} &\leq EPS(1 + EPS/2)(|x.\mathrm{re} \otimes y.\mathrm{re}| \oplus |x.\mathrm{im} \otimes y.\mathrm{im}|) \\ &+EPS(1 + EPS/2)(|x.\mathrm{re} \otimes y.\mathrm{im}| \oplus |x.\mathrm{im} \otimes y.\mathrm{re}|) \\ &\leq EPS \otimes ((1 + 2EPS) \otimes ((|x.\mathrm{re} \otimes y.\mathrm{re}| \oplus |x.\mathrm{im} \otimes y.\mathrm{im}|) \\ &\oplus(|x.\mathrm{re} \otimes y.\mathrm{im}| \oplus |x.\mathrm{im} \otimes y.\mathrm{re}|))). \end{aligned}$$

□

Proposition 6.19. (D/X) *If x is a double and y is an XComplex, then $S(x/y) \supseteq S(x)/S(y)$, where*

$$\begin{aligned} x/y &\equiv (\mathrm{re}, \mathrm{im}; e) \text{ with} \\ re &= (x \otimes y.\mathrm{re}) \oslash nrm \text{ where } nrm = y.\mathrm{re} \otimes y.\mathrm{re} \oplus y.\mathrm{im} \otimes y.\mathrm{im} \\ \mathrm{im} &= -(x \otimes y.\mathrm{im}) \oslash nrm \\ e &= (2EPS) \otimes ((1 + 2EPS) \otimes (|\mathrm{re}| \oplus |\mathrm{im}|)). \end{aligned}$$

Proof. The true version of x/y is equal to

$$(x \times y.\mathrm{re} + i(-x \times y.\mathrm{im}))/((y.\mathrm{re})^2 + (y.\mathrm{im})^2)$$

and we need to compare this with the machine version to find the error. Further, this error is less than or equal to the sum of the real error and the imaginary error. Thus, we start with the real calculation (as in the statement of the proposition, we use nrm to represent the machine version of $(y.\mathrm{re})^2 + (y.\mathrm{im})^2$).

$$\begin{aligned} &\left| \frac{x \times y.\mathrm{re}}{(y.\mathrm{re})^2 + (y.\mathrm{im})^2} - ((x \otimes y.\mathrm{re}) \oslash nrm) \right| \\ &\quad \leq \left| (x \otimes y.\mathrm{re}) \oslash nrm - \frac{x \otimes y.\mathrm{re}}{nrm} \right| \\ &\qquad + \left| \frac{x \otimes y.\mathrm{re}}{nrm} - \frac{x \times y.\mathrm{re}}{nrm} \right| + \left| \frac{x \times y.\mathrm{re}}{nrm} - \frac{x \times y.\mathrm{re}}{(y.\mathrm{re})^2 + (y.\mathrm{im})^2} \right|. \end{aligned}$$

Before continuing, let us compare $\frac{1}{nrm}$ and $\frac{1}{(y.\mathrm{re})^2+(y.\mathrm{im})^2}$ by developing a formula for comparing $\frac{1}{a^2+b^2}$ and its associated $\frac{1}{nrm}$:

Lemma 6.20.

$$\left| \frac{1}{nrm} - \frac{1}{a^2 + b^2} \right| \leq (EPS + (EPS/2)^2) \frac{1}{nrm}$$

where $nrm = a \otimes a \oplus b \otimes b$.

Proof. We compute that

$$\left(\frac{1}{1 + EPS/2} \right)^2 \times nrm \leq a^2 + b^2 \leq \left(\frac{1}{1 - EPS/2} \right)^2 \times nrm;$$

hence

$$\frac{1}{nrm}(1 - EPS/2)^2 \leq \frac{1}{a^2 + b^2} \leq \frac{1}{nrm}(1 + EPS/2)^2.$$

It then follows that

$$\begin{aligned} \left| \frac{1}{nrm} - \frac{1}{a^2 + b^2} \right| &\leq \frac{1}{nrm}(1 + EPS/2)^2 - \frac{1}{nrm} \\ &= \frac{1}{nrm}((1 + EPS/2)^2 - 1) = (EPS + (EPS/2)^2) \frac{1}{nrm}. \end{aligned}$$

□

Getting back to our main calculation (with $nrm = y.\mathrm{re} \otimes y.\mathrm{re} \oplus y.\mathrm{im} \otimes y.\mathrm{im}$), we have

$$\begin{aligned}
&\left|(x \otimes y.\mathrm{re}) \oslash nrm - \frac{x \otimes y.\mathrm{re}}{nrm}\right| \\
&\quad + \left|\frac{x \otimes y.\mathrm{re}}{nrm} - \frac{x \times y.\mathrm{re}}{nrm}\right| + \left|\frac{x \times y.\mathrm{re}}{nrm} - \frac{x \times y.re}{(y.\mathrm{re})^2 + (y.\mathrm{im})^2}\right| \\
&\leq (EPS/2)\frac{|x \otimes y.\mathrm{re}|}{nrm} + (EPS/2)\frac{|x \otimes y.\mathrm{re}|}{nrm} + (EPS + (EPS/2)^2)\frac{|x \times y.\mathrm{re}|}{nrm} \\
&= (EPS/2)\left(\frac{1}{nrm}\right)(2|x \otimes y.\mathrm{re}| + (2 + EPS/2) \times |x \times y.\mathrm{re}|) \\
&\leq (EPS/2)\left(\frac{1}{nrm}\right)(2|x \otimes y.\mathrm{re}| + (2 + EPS/2)(1 + EPS/2) \times |x \otimes y.\mathrm{re}|) \\
&\quad = (EPS/2)\left(\frac{1}{nrm}\right)(|x \otimes y.\mathrm{re}|)(2 + (2 + EPS/2)(1 + EPS/2)) \\
&\leq (EPS/2)(4 + 3EPS/2 + (EPS/2)^2)(|x \otimes y.\mathrm{re}|)\left(\frac{1}{nrm}\right) \\
&\leq (EPS/2)(4 + 3EPS/2 + (EPS/2)^2)(1 + EPS/2)(|x \otimes y.\mathrm{re}| \oslash nrm) \\
&\leq (2EPS)(1 + 3EPS/8 + (EPS/4)^2)(1 + EPS/2)(|(x \otimes y.re \oslash nrm)|).
\end{aligned}$$

We also get the analogous formula for the imaginary contribution for the error, so our total error is bounded by

$$\begin{aligned}
&(2EPS)(1 + 3EPS/8 + (EPS/4)^2)(1 + EPS/2)((|(x \otimes y.\mathrm{re}) \oslash nrm|) \\
&\qquad +(|(x \otimes y.\mathrm{im}) \oslash nrm|)) \\
&\quad \leq (2EPS)(1 + 3EPS/8 + (EPS/4)^2)(1 + EPS/2)^2 \\
&\qquad \cdot ((|(x \otimes y.\mathrm{re}) \oslash nrm|) \oplus (|(x \otimes y.\mathrm{im}) \oslash nrm|)) \\
&\quad \leq (2EPS)(1 - EPS/2)(1 + 2EPS) \\
&\qquad \cdot ((|(x \otimes y.\mathrm{re}) \oslash nrm|) \oplus (|(x \otimes y.\mathrm{im}) \oslash nrm|)) \\
&\quad \leq (2EPS) \otimes ((1 + 2EPS) \otimes ((|(x \otimes y.\mathrm{re}) \oslash nrm|) \\
&\qquad \oplus(|(x \otimes y.\mathrm{im}) \oslash nrm|))).
\end{aligned}$$

Here we used the fact that

$$(1 + 3EPS/8 + (EPS/4)^2)(1 + EPS/2)^2 \leq (1 - EPS/2)(1 + 2EPS). \qquad \square$$

This should give the flavor of division proofs. As such, we will skip the proofs of X/X and A/A and simply refer to the *A*nnals web site.

Proposition 6.21. *(X/X) If x and y are XComplexes, then $S(x/y) \supseteq S(x)/S(y)$, where*

$$\begin{aligned} x/y &\equiv (\mathrm{re}, \mathrm{im}; e) \text{ with} \\ \mathrm{re} &= (x.\mathrm{re} \otimes y.\mathrm{re} \oplus x.\mathrm{im} \otimes y.\mathrm{im}) \oslash nrm \\ &\quad \text{where } nrm = y.\mathrm{re} \otimes y.\mathrm{re} \oplus y.\mathrm{im} \otimes y.\mathrm{im} \\ \mathrm{im} &= (x.\mathrm{im} \otimes y.\mathrm{re} \ominus x.\mathrm{re} \otimes y.\mathrm{im}) \oslash nrm \\ e &= (5EPS/2) \otimes ((1 + 3EPS) \otimes A) \text{ where} \\ A &= ((|x.\mathrm{re} \otimes y.\mathrm{re}| \oplus |x.\mathrm{im} \otimes y.\mathrm{im}|) \oplus (|x.\mathrm{im} \otimes y.\mathrm{re}| \oplus |x.\mathrm{re} \otimes y.\mathrm{im}|)) \oslash nrm. \end{aligned}$$

Proposition 6.22. *(A/A) If x and y are AComplexes with $y.e < 100EPS \otimes |y|$, or, more accurately,*

$$(y.e)^2 < ((10000EPS) \otimes EPS) \otimes nrm$$

then $S(x/y) \supseteq S(x)/S(y)$, where

$$\begin{aligned} x/y &\equiv (\mathrm{re}, \mathrm{im}; e) \text{ with} \\ \mathrm{re} &= (x.\mathrm{re} \otimes y.\mathrm{re} \oplus x.\mathrm{im} \otimes y.\mathrm{im}) \oslash nrm \\ &\quad \text{where } nrm = y.\mathrm{re} \otimes y.\mathrm{re} \oplus y.\mathrm{im} \otimes y.\mathrm{im} \\ \mathrm{im} &= (x.\mathrm{im} \otimes y.\mathrm{re} \ominus x.\mathrm{re} \otimes y.\mathrm{im}) \oslash nrm \\ e &= (1 + 4EPS) \otimes (((5EPS/2) \otimes A \oplus (1 + 103EPS) \otimes B) \oslash nrm) \text{ where} \\ A &= (|x.\mathrm{re} \otimes y.\mathrm{re}| \oplus |x.\mathrm{im} \otimes y.\mathrm{im}|) \oplus (|x.\mathrm{im} \otimes y.\mathrm{re}| \oplus |x.\mathrm{re} \otimes y.\mathrm{im}|) \\ B &= x.e \otimes (|y.\mathrm{re}| \oplus |y.\mathrm{im}|) \oplus (|x.\mathrm{re}| \oplus |x.\mathrm{im}|) \otimes y.e. \end{aligned}$$

In our last proposition we will construct the square-root function. As a warm-up, ignoring round-off error, our construction is as follows. If $x = x.re + ix.im$ then $\sqrt{x} = s + \mathrm{i}d$ where $s = \sqrt{(|x.\mathrm{re}| + h(x.\mathrm{re}, x.\mathrm{im}))/2}$ and $d = x.\mathrm{im}/(2s)$ when $x.re > 0.0$, and $\sqrt{x} = d + is$ otherwise. Thus, we take our (no-round-off) square roots to be in the first and fourth quadrants.

Proposition 6.23. *($\sqrt{X}$) If x is an XComplex, then $S(\sqrt{x}) \supseteq \sqrt{S(x)}$ where we let $s_o = \sqrt[o]{(|x.\mathrm{re}| \oplus h_o(x.\mathrm{re}, x.\mathrm{im})) \otimes 0.5}$ and $d_o = (x.\mathrm{im} \oslash s) \otimes 0.5$, and define*

$$\begin{aligned} \sqrt{x} &\equiv (\mathrm{re}, \mathrm{im}; e) \text{ where} \\ \mathrm{re} &= s_o \text{ if } x.\mathrm{re} > 0.0 \text{ and } re = d_o \text{ otherwise,} \\ \mathrm{im} &= d_o \text{ if } x.\mathrm{re} > 0.0 \text{ and } im = s_o \text{ otherwise,} \\ e &= EPS \otimes ((1 + 4EPS) \otimes (1.25 \otimes s_o \oplus 1.75 \otimes |d_o|)). \end{aligned}$$

Proof. This will be a little nasty. Let us begin by analyzing e_s, which is the difference between the true calculation of s and the machine calculation of s, that is $e_s = |s - s_o|$.

First, we bound s.

$$\begin{aligned} s &= \sqrt{(|x.\mathrm{re}| + h(x.\mathrm{re}, x.\mathrm{im})) * 0.5} \\ &\leq (1+EPS)^{1/2}\sqrt{(|x.\mathrm{re}| + h_o(x.\mathrm{re}, x.\mathrm{im})) * 0.5} \\ &\leq (1+EPS)^{1/2}(1+EPS/2)^{1/2}\sqrt{(|x.\mathrm{re}| \oplus h_o(x.\mathrm{re}, x.\mathrm{im})) * 0.5} \\ &\leq (1+EPS)^{1/2}(1+EPS/2)^{1/2} \\ &\quad \cdot (1+EPS/2)\sqrt[o]{(|x.\mathrm{re}| \oplus h_o(x.\mathrm{re}, x.\mathrm{im})) * 0.5} \\ &= (1+EPS)^{1/2}(1+EPS/2)^{3/2}s_o. \end{aligned}$$

By a power series expansion, we see that

$$\begin{aligned} (1+EPS)^{1/2}(1+EPS/2)^{3/2} &= \left(1 + \frac{1}{2}EPS - \frac{1}{8}EPS^2 + \cdots\right) \\ &\quad + \left(1 + \frac{3}{2}EPS/2 + \frac{3}{8}(EPS/2)^2 + \cdots\right) \\ &= \left(1 + \frac{5}{4}EPS + \frac{11}{32}EPS^2 + \cdots\right), \end{aligned}$$

so that,

$$s \leq \left(1 + \frac{5}{4}EPS + \frac{11}{32}EPS^2 + \cdots\right) s_o.$$

Similarly,

$$s \geq \left(1 - \frac{5}{4}EPS\right) s_o.$$

Thus, we can bound the s error,

$$\begin{aligned} e_s = |s - s_o| &\leq \left(\left(1 + \frac{5}{4}EPS + \frac{11}{32}EPS^2 + \cdots\right) - 1\right) s_o \\ &= \left(\frac{5}{4}EPS + \frac{11}{32}EPS^2 + \cdots\right) s_o. \end{aligned}$$

Next, we analyze e_d, which is the absolute value of the difference between the true

calculation of d and the machine calculation of d. That is, $e_d = |d - d_o|$.

$$
\begin{aligned}
e_d &= |x.\mathrm{im}/(2s) - x.\mathrm{im} \oslash (2s_o)| \\
&\leq |x.\mathrm{im} \oslash (2s_o) - x.\mathrm{im}/(2s_o)| + |x.\mathrm{im}/(2s_o) - x.\mathrm{im}/(2s)| \\
&\leq (EPS/2)|x.\mathrm{im}/(2s_o)| + \left| \frac{x.\mathrm{im}}{2} \frac{s - s_o}{ss_o} \right| \\
&\leq (EPS/2)|x.\mathrm{im}/(2s_o)| + \left| \frac{x.\mathrm{im}}{2} \frac{1}{ss_o} ((5/4)EPS + (11/32)EPS^2 + \cdots)s_o \right| \\
&\leq (EPS/2)|x.\mathrm{im}/(2s_o)| \\
&\quad + \left| \frac{x.\mathrm{im}}{2} \frac{1}{s_o(1 - (5/4)EPS)} ((5/4)EPS + (11/32)EPS^2 + \cdots) \right| \\
&= (EPS/2)|x.\mathrm{im}/(2s_o)| \left(1 + \frac{(5/2) + (11/16)EPS + \cdots)}{(1 - (5/4)EPS)} \right) \\
&= (EPS/2) \frac{(7/2) + (-9/16)EPS + \cdots}{(1 - (5/4)EPS)} |x.\mathrm{im}/(2s_o)| \\
&\leq (EPS/2)(1 + EPS/2) \frac{7/2}{(1 - (5/4)EPS)} |x.\mathrm{im} \oslash (2s_o)| \\
&= (EPS/2)(1 + EPS/2) \frac{7/2}{(1 - (5/4)EPS)} |d_o|.
\end{aligned}
$$

Finally, we can bound the overall error $e = e_s + e_d$.

$$\begin{aligned}
e_s + e_d &\leq \left(\frac{5}{4}EPS + \frac{11}{32}EPS^2 + \cdots\right) s_o \\
&\quad + (EPS/2)(1 + EPS/2)\frac{7/2}{(1 - (5/4)EPS)}|d_o| \\
&\leq \left(EPS + \frac{11}{40}EPS^2 + \cdots\right)\left(\frac{5}{4}s_o\right) \\
&\quad + EPS(1 + EPS/2)\frac{1}{(1 - (5/4)EPS)}\left|\frac{7}{4}d_o\right| \\
&\leq EPS(1 + EPS/2)\frac{1}{(1 - (5/4)EPS)}\left(\frac{5}{4}s_o\right) \\
&\quad + EPS(1 + EPS/2)\frac{1}{(1 - (5/4)EPS)}\left|\frac{7}{4}d_o\right| \\
&\leq EPS(1 + EPS/2)\frac{1}{(1 - (5/4)EPS)}\left(\frac{5}{4}s_o + \left|\frac{7}{4}d_o\right|\right) \\
&\leq EPS(1 + EPS/2)^3\frac{1}{(1 - (5/4)EPS)}\left(\frac{5}{4} \otimes s_o \oplus \left|\frac{7}{4} \otimes d_o\right|\right) \\
&\leq EPS(1 - (EPS/2))(1 + 4EPS)\left(\frac{5}{4} \otimes s_o \oplus \left|\frac{7}{4} \otimes d_o\right|\right) \\
&\leq EPS \otimes \left((1 + 4EPS) \otimes \left(\frac{5}{4} \otimes s_o \oplus \left|\frac{7}{4} \otimes d_o\right|\right)\right).
\end{aligned}$$

□

Now, we develop two formulas for the absolute value of an XComplex.

Formula 6.24. (a$bsUB(X)$) If x is an XComplex, then there is an upper bound on the absolute value of x as follows:

$$\begin{aligned}
|x| &= h(x.\mathrm{re}, x.\mathrm{im}) \leq (1 + EPS)h_\circ(x.\mathrm{re}, x.\mathrm{im}) \\
&\leq (1 - EPS/2)(1 + 2EPS)h_\circ(x.\mathrm{re}, x.\mathrm{im}) \\
&\leq (1 + 2EPS) \otimes h_\circ(x.\mathrm{re}, x.\mathrm{im}).
\end{aligned}$$

Thus, we define

$$\mathrm{a}bsUB(x) = (1 + 2EPS) \otimes h_\circ(x.\mathrm{re}, x.\mathrm{im}).$$

Formula 6.25. (a$bsLB(X)$). If x is an XComplex, then we get a lower bound on the absolute value of x as follows.

$$\begin{aligned}
|x| &= h(x.\mathrm{re}, x.\mathrm{im}) \geq (1 - EPS)h_\circ(x.\mathrm{re}, x.\mathrm{im}) \\
&\geq (1 + EPS/2)(1 - 2EPS)h_\circ(x.\mathrm{re}, x.\mathrm{im}) \\
&\geq (1 - 2EPS) \otimes h_\circ(x.\mathrm{re}, x.\mathrm{im}).
\end{aligned}$$

Thus, we define

$$\mathrm{a}bsLB(x) = (1 - 2EPS) \otimes h_\circ(x.\mathrm{re}, x.\mathrm{im}).$$

Finally, in several places in the *v*erify program we perform a standard operation on a pair of doubles and must take into account round-off error. This is easy if we use Lemma 6.6.

For example, in *i*nequalityHolds we want to show that

$$wh \times wh > \mathrm{a}bsUB(along),$$

where $wh = \mathrm{a}bsLB(whirle)$. By Lemma 6.6, we know that

$$(1 - EPS) \otimes (wh \otimes wh) \leq wh \times wh$$

and we simply test that

$$(1 - EPS) \otimes (wh \otimes wh) \geq \mathrm{a}bsUB(along).$$

A slightly more complicated version of this occurs in the computer calculation of p$os[i]$ and $size[i]$, that is, the center and size of a sub-box. Prior to multiplication by s$cale[i] = 2^{(5-i)/6}$, the calculations of pos and $size$ are exact. However, multiplication by $scale$ introduces round-off error. For the center of the sub-box we will have the computer use p$os[i] \otimes$ s$cale[i]$ with the realization that this is not necessarily p$os[i] \times$ s$cale[i]$. Thus, we have to choose appropriate sizes to ensure that the machine sub-box contains the true sub-box.

Notationally, this is annoying, because we typically use a computer command like p$os[i] =$ p$os[i] \otimes$ s$cale[i]$, while in an exposition, we need to avoid that. We will denote the true center of the sub-box by $p[i]$ and the machine center of the sub-box by $p_0[i]$, and the true and machine sizes will be denoted $s[i]$ and $s_0[i]$. We will let $pos[i]$ and $size[i]$ be the position and size (true and machine are the same) before multiplication by s$cale[i]$.

Let $p[i] = pos[i] \times$ s$cale[i]$, $p_0[i] = pos[i] \otimes$ s$cale[i]$, and $s[i] = size[i] \times$ s$cale[i]$. We must select $s_0[i]$ so that $p_0[i] + s_0[i] \geq p[i] + s[i]$. (Here, taking $+$ on the left-hand side is correct, because the need for machine calculation there is incorporated at other points in the programs.) So, we must find $s_0[i]$ such that $s_0[i] \geq (p[i] - p_0[i]) + s[i]$.

$$\begin{aligned}
(p[i] - p_0[i]) + s[i]. &\leq (EPS/2)|p_0[i]| + \mathrm{s}ize[i] \times \mathrm{s}cale[i] \\
&\leq (EPS/2)|p_0[i]| + (1 + EPS/2)(\mathrm{s}ize[i] \otimes \mathrm{s}cale[i]) \\
&\leq (1 + EPS/2)((EPS/2)|p_0[i]| + (\mathrm{s}ize[i] \otimes \mathrm{s}cale[i])) \\
&\leq (1 + EPS/2)^2((EPS/2)|p_0[i]| \oplus (\mathrm{s}ize[i] \otimes \mathrm{s}cale[i])) \\
&\leq (1 + 2EPS) \otimes ((EPS/2)|p_0[i]| \oplus (\mathrm{s}ize[i] \otimes \mathrm{s}cale[i])).
\end{aligned}$$

Thus we take

$$s_0[i] = (1 + 2EPS) \otimes ((EPS/2)|p_0[i]| \oplus (\mathrm{s}ize[i] \otimes \mathrm{s}cale[i])).$$

This also works to give $p_0[i] - s_0[i] \leq p[i] - s[i]$. □

Chapter 7

AffApproxes with round-off error

In Chapter 5, we saw how to do calculations with AffApproxes. Here, we incorporate round-off error into these calculations.

Conventions 7.1. An AffApprox x is a five-tuple $(x.f; x.f_0, x.f_1, x.f_2; x.err)$ consisting of four complex numbers $(x.f, x.f_0, x.f_1, x.f_2)$ and one real number $x.err$. In Chapter 5 , the real number was denoted $x.e$, but it seems preferable to use $x.err$ here. Recall (Definition 5.4 that an AffApprox x represents the set $S(x)$ of functions from $A = \{(z_0, z_1, z_2) \in \mathbf{C}^3 : |z_k| \leq 1 \text{ for } k \in \{0, 1, 2\}\}$ to $\mathbf{C}$ that are $x.err$-well-approximated by the affine function $x.f + x.f_0 z_0 + x.f_1 z_1 + x.f_2 z_2$.

Remarks 7.2. A review of Definition 6.2 (XComplexes and AComplexes; loosely, exact and approximate complex numbers) might be helpful at this time.

One approach to round-off error for AffApproxes would be to replace the four complex numbers in the definition of AffApprox by four AComplex numbers complete with their round-off errors, and similarly for the one real number. We will not do this because it would necessitate keeping track of five separate round-off-error terms when we do AffApprox calculations.

Instead, we will replace the four complex numbers by four XComplexes and push all the round-off error into the $.err$ term. Thus, the definition of AffApprox remains essentially the same as in 5 We note that, in doing an AffApprox calculation, our subsidiary calculations will generally be on XComplex numbers and produce an AComplex number whose $.e$ term will be plucked off and forced into the $.err$ term of the final AffApprox.

Conventions 7.3. i) In what follows, we will use Basic Properties 6.5 and Lemmas 6.6 and 6.7. Also, the corresponding propositions in Chapter 5 will be utilized. (for example, Proposition 5.13 corresponds to Proposition 7.10.

ii) Some notational simplifications will be introduced: d$ist(x)$ before Proposition 7.9, ax before Propositions 7.11, 7.12, 7.14, (the middle usage of ax differs slightly from the other two), and ay before Propositions 7.11, 7.12.

iii) We will try to keep our notation fairly consistent with that of the $v$erify computer

program, and this will produce some mildly peculiar notation. In particular, in the operations pertaining to Propositions 7.5 and beyond, the resultant AffApproxes will be denoted $(r\_f.z; r\_f_1.z, r\_f_2.z, r\_f_3.z; r\_error)$ where the first four terms are XComplexes and the last term is a double (technically, $r\_f.z$ is the XComplex part of the AComplex r_f and similarly for the $r\_f_k.z$ terms). One break with the notation of the programs though is that AffApproxes are called (in the programs) ACJ's, which stands for "Approximate Complex 1-Jets."

iv) The propositions that follow include in their statements the definitions of the various operations on AffApproxes (see Remark 5.6 iii).

v) We remind the reader (see Conventions 6.4 that all machine operations act on machine numbers, and that the various variables appearing in the propositions are assumed to be doubles.

$-X$:

Proposition 7.4. *If x is an AffApprox then $S(-x) = -(S(x))$ where*

$$-x \equiv (-x.f; -x.f_0, -x.f_1, -x.f_2; x.err).$$

$X + Y$: We analyze the addition of the AffApproxes $x = (x.f; x.f_0, x.f_1, x.f_2; x.err)$ and $y = (y.f; y.f_0, y.f_1, y.f_2; y.err)$. To get the first term in $x+y$ we add the XComplex numbers $x.f$ and $y.f$; which produce the AComplex number $r\_f = x.f + y.f$ (see Proposition 6.12), and then we pluck off the XComplex part, which we denote $r\_f.z$. The round-off error part $r\_f.e$ will be foisted into the overall error term r_error for $x+y$. Similarly for the next three terms in $x + y$.

Abstractly, the overall error term r_error comes from adding the round-off error contributions $r\_f.e,\ r\_f_0.e,\ r\_f_1.e,\ r\_f_2.e$ and the AffApprox error contributions $x.err,\ y.err$. Of course, we have to produce a machine version.

Proposition 7.5. *If x and y are AffApproxes, then $S(x + y) \supseteq S(x) + S(y)$, where*

$$x + y \equiv (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error)$$

with

$$\begin{aligned} r\_f &= x.f + y.f, \\ r\_f_k &= x.f_k + y.f_k, \\ r\_error &= (1 + 3EPS) \\ &\quad \otimes ((x.err \oplus y.err) \oplus ((r\_f.e \oplus r\_f_0.e) \oplus (r\_f_1.e \oplus r\_f_2.e))). \end{aligned}$$

Proof. The error is given by

$$\begin{aligned}
&(x.err + y.err) + \ ((r_f.e + r_f_0.e) + (r_f_1.e + r_f_2.e)) \\
&\leq \ (1 + EPS/2)(x.\text{err} \oplus y.\text{err}) \\
&\quad + (1 + EPS/2)((r_f.e \oplus r_f_0.e) + (r_f_1.e \oplus r_f_2.e)) \\
&\leq \ (1 + EPS/2)^3((x.\text{err} \oplus y.\text{err}) \oplus ((r_f.e \oplus r_f_0.e) \oplus (r_f_1.e \oplus r_f_2.e))) \\
&\leq \ (1 + 3EPS) \otimes ((x.\text{err} \oplus y.\text{err}) \oplus ((r_f.e \oplus r_f_0.e) \oplus (r_f_1.e \oplus r_f_2.e))).
\end{aligned}$$

To get the last line we used Lemma 6.7. □

$X - Y$:

Proposition 7.6. *If x and y are AffApproxes, then $S(x - y) \supseteq S(x) - S(y)$, where*

$$x - y \equiv (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error)$$

with

$$\begin{aligned}
r_f &= x.f - y.f, \\
r_f_k &= x.f_k - y.f_k, \\
r_error &= (1 + 3EPS) \\
&\quad \otimes ((x.err \oplus y.err) \oplus ((r_f.e \oplus r_f_0.e) \oplus (r_f_1.e \oplus r_f_2.e))).
\end{aligned}$$

$X + D$: Here, we add the AffApprox $x = (x.f; x.f_0, x.f_1, x.f_2; x.\text{err})$ to the double y. The only terms that change are the first and the last.

Proposition 7.7.*If x is an AffApprox and y is a double, then $S(x+y) \supseteq S(x)+S(y)$, where*

$$x + y \equiv (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error)$$

with

$$\begin{aligned}
r_f &= x.f + y, \\
r_f_k &= x.f_k, \\
r_error &= (1 + EPS) \otimes (x.err \oplus r_f.e).
\end{aligned}$$

Proof. The error is given by

$$x.err + r\_f.e \leq (1 + EPS) \otimes (x.\text{err} \oplus r\_f.e)$$

by Lemma 6.6. □

$X - D$:

Proposition 7.8. *If x is an AffApprox and y is a double, then $S(x-y) \supseteq S(x)-S(y)$, where*

$$x - y \equiv (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error)$$

with

$$\begin{aligned} r\_f &= x.f - y, \\ r\_f_k &= x.f_k, \\ r\_error &= (1 + EPS) \otimes (x.err \oplus r\_f.e). \end{aligned}$$

$X \times Y$: We multiply the AffApproxes x and y while pushing all error into the .err term.

We will use the functions (see Formulas 6.24 and 6.25, at the end of Chapter 6) $absUB = (1 + 2EPS) \otimes hypot_o(x.re, x.im)$ and $absLB(x) = (1 - 2EPS) \otimes hypot_o(x.re, x.im)$.

When x is an AffApprox, we define $dist(x)$ to be

$$(1 + 2EPS) \otimes (absUB(x.f_0) \oplus (absUB(x.f_1) \oplus absUB(x.f_2))).$$

This is the machine representation of the sum of the absolute values of the linear terms in the AffApprox x (the proof is straightforward).

Proposition 7.9. *If x and y are AffApproxes, then $S(x \times y) \supseteq S(x) \times S(y)$, where*

$$x \times y \equiv (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error)$$

with

$$\begin{aligned} r\_f &= x.f \times y.f, \\ r\_f_k &= x.f \times y.f_k + x.f_k \times y.f, \\ r\_error &= (1 + 3EPS) \otimes (A \oplus (B \oplus C)), \end{aligned}$$

and

$$\begin{aligned} A &= (dist(x) \oplus x.err) \otimes (dist(y) \oplus y.err), \\ B &= absUB(x.f) \otimes y.err \oplus absUB(y.f) \otimes x.err, \\ C &= (r\_f.e \oplus r\_f_0.e) \oplus (r\_f_1.e \oplus r\_f_2.e). \end{aligned}$$

Proof. We add the non-round-off error term for $x \times y$ to the various round-off error terms that accumulated.

$$\begin{aligned} &((dist(x) + x.\mathrm{err}) \times (dist(y) + y.\mathrm{err})) + ((absUB(x.f) \times y.\mathrm{err} \\ &\quad + absUB(y.f) \times x.\mathrm{err}) + (r\_f.e + r\_f_0.e) + (r\_f_1.e + r\_f_2.e)) \\ &\leq (1 + EPS/2)^3[(dist(x) \oplus x.\mathrm{err}) \otimes (dist(y) \oplus y.\mathrm{err})] \\ &\quad + (1 + EPS/2)^2\{(absUB(x.f) \otimes y.\mathrm{err} \\ &\quad \oplus absUB(y.f) \otimes x.\mathrm{err}) + ((r\_f.e \oplus r\_f_0.e) \oplus (r\_f_1.e \oplus r\_f_2.e))\} \\ &\leq (1 + EPS/2)^3 A + (1 + EPS/2)^3 (B \oplus C) \\ &\leq (1 + 3EPS) \otimes (A \oplus (B \oplus C)). \end{aligned}$$

□

$X \times D$:

Proposition 7.10. *If x is an AffApprox and y is a double, then $S(x \times y) \supseteq S(x) \times S(y)$, where*

$$x \times y = (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error)$$

with

$$\begin{aligned} r\_f &= x.f \times y, \\ r\_f_k &= x.f_k \times y, \\ r\_error &= (1 + 3EPS) \\ &\quad \otimes ((x.err \otimes |y|) \oplus ((r\_f.e \oplus r\_f_0.e) \oplus (r\_f_1.e \oplus r\_f_2.e))). \end{aligned}$$

X/Y: For convenience, let $ax = absUB(x.f),\ ay = absLB(y.f)$.

Proposition 7.11. *If x and y are AffApproxes with $D > 0$ (see below), then $S(x/y) \supseteq S(x)/S(y)$, where*

$$x/y \equiv (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error)$$

with

$$\begin{aligned} r\_f &= x.f/y.f, \\ r\_f_k &= (x.f_k \times y.f - x.f \times y.f_k)/(y.f \times y.f), \\ r.error &= (1 + 3EPS) \otimes (((1 + 3EPS) \otimes A \ominus (1 - 3EPS) \otimes B) \oplus C), \end{aligned}$$

and

$$\begin{aligned} A &= (ax \oplus (dist(x) \oplus x.err)) \oslash D, \\ B &= (ax \oslash ay \oplus dist(x) \oslash ay) \oplus ((dist(y) \otimes ax) \oslash (ay \otimes ay)), \\ C &= (r\_f.e \oplus r\_f_0.e) \oplus (r\_f_1.e \oplus r\_f_2.e), \\ D &= ay \ominus (1 + EPS) \otimes (dist(y) \oplus y.err). \end{aligned}$$

Proof. As usual, we add the round-off errors to the old AffApprox error, taking into account round-off error, working on it bit by bit.

$$\begin{aligned} &(ax + \mathrm{d}ist(x) + x.\mathrm{err})/(ay - (\mathrm{d}ist(y) + y.\mathrm{err})) \\ &\quad \le (1 + EPS/2)^2 \\ &\qquad \times (ax \oplus (\mathrm{d}ist(x) \oplus x.\mathrm{err}))/(ay - (1 + EPS) \otimes (\mathrm{d}ist(y) \oplus y.\mathrm{err})) \\ &\quad \le (1 + EPS/2)^2 (ax \oplus (\mathrm{d}ist(x) \oplus x.\mathrm{err}))/\Big(\left(\frac{1}{1 + EPS/2} \right) \\ &\qquad \times (ay \ominus (1 + EPS) \otimes (\mathrm{d}ist(y) \oplus y.\mathrm{err})) \Big) \\ &\quad \le (1 + EPS/2)^4 (ax \oplus (\mathrm{d}ist(x) \oplus x.\mathrm{err})) \\ &\qquad \oslash (ay \ominus (1 + EPS) \otimes (\mathrm{d}ist(y) \oplus y.\mathrm{err})) \\ &\quad \le (1 + 3EPS) \otimes A. \end{aligned}$$

The next term, being subtracted, requires opposite inequalities.

$$\begin{aligned}
&(ax/ay + \mathrm{d}ist(x)/ay) + \mathrm{d}ist(y) \times ax/(ay \times ay) \\
&\quad \geq (1 - EPS/2)(ax \oslash ay + \mathrm{d}ist(x) \oslash ay) \\
&\qquad +(1 - EPS/2)(\mathrm{d}ist(y) \otimes ax)/(\frac{1}{1 - EPS/2})(ay \otimes ay) \\
&\quad \geq ((1 - EPS/2)^4((ax \oslash ay \oplus \mathrm{d}ist(x) \oslash ay) \oplus ((\mathrm{d}ist(y) \otimes ax) \oslash (ay \otimes ay))) \\
&\quad \geq (1 + EPS/2)(1 - 3EPS)(B) \geq (1 - 3EPS) \otimes B.
\end{aligned}$$

Finally, we do the round-off terms

$$((r\_f.e + r\_f_0.e) + (r\_f_1.e + r\_f_2.e)) \leq (1 + EPS/2)^2 C$$

and we put these three pieces together:

$$\begin{aligned}
&(ax + \mathrm{d}ist(x) + x.err)/(ay - (\mathrm{d}ist(y) + y.err)) \\
&\qquad -((ax/ay + \mathrm{d}ist(x)/ay) + \mathrm{d}ist(y) \times ax/(ay \times ay)) \\
&\qquad +((r_f.e + r_f_0.e) + (r_f_1.e + r_f_2.e)) \\
&\quad \leq (1 + 3EPS) \otimes A - (1 - 3EPS) \otimes B + (1 + EPS/2)^2 C \\
&\quad \leq (1 + EPS/2)^2(((1 + 3EPS) \otimes A \ominus (1 - 3EPS) \otimes B) + C) \\
&\quad \leq (1 + 3EPS) \otimes (((1 + 3EPS) \otimes A \ominus (1 - 3EPS) \otimes B) \oplus C).
\end{aligned}$$

□

D/X: We divide a double x by an AffApprox y. For convenience, let $ax = |x|$, $ay = \mathrm{a}bsLB(y.f)$. Having done division out in the previous proposition, we will skip the proof of Proposition 7.12. See the *A*nnals web site for the proof.

Proposition 7.12. *If x is a double and y is an AffApprox with $D > 0$ (see below), then $S(x/y) \supseteq S(x)/S(y)$, where*

$$x/y \equiv (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error)$$

with

$$\begin{aligned}
r_f &= x/y.f, \\
r_f_k &= -(x \times y.f_k)/(y.f \times y.f), \\
r_error &= (1 + 3EPS) \otimes (((1 + 2EPS) \otimes (ax \oslash D) \ominus (1 - 3EPS) \otimes B) \oplus C), \\
B &= ax \oslash ay \oplus (dist(y) \otimes ax \oslash (ay \otimes ay)), \\
C &= (r_f.e \oplus r_f_0.e) \oplus (r_f_1.e \oplus r_f_2.e), \\
D &= ay \ominus (1 + EPS) \otimes (dist(y) \oplus y.err).
\end{aligned}$$

X/D: We divide an AffApprox x by a nonzero double y (the computer will object if $y = 0$). The proof is easy and so we delete it.

Proposition 7.13. *If x is an AffApprox and y is a nonzero double, then $S(x/y) \supseteq S(x)/S(y)$, where*

$$x/y \equiv (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error),$$

with

$$\begin{aligned} r\_f &= x.f/y \\ r\_f_k &= x.f_k/y \\ r\_error &= (1+3EPS) \otimes ((x.err \oslash |y|) \oplus [(r\_f.e \oplus r\_f_0.e) \oplus (r\_f_1.e \oplus r\_f_2.e)]). \end{aligned}$$

$\sqrt{X}$: Here, x is an AffApprox and we let $ax = absUB(x.f)$. There are two cases to consider depending on whether or not $D = ax \ominus (1+EPS) \otimes (dist(x) \oplus x.err)$ is or is not greater than zero.

Proposition 7.14. *If x is an AffApprox and $D = ax \ominus (1+EPS) \otimes (dist(x) \oplus x.err)$ is not greater than zero, then $S(\sqrt{x}) \supseteq \sqrt{S(x)}$, where we use the crude overestimate*

$$\sqrt{x} \equiv \left(0; 0, 0, 0; (1+2EPS) \otimes \sqrt[o]{(ax \oplus (x\ dist \oplus x.err))}\right).$$

Proof.

$$\begin{aligned} \sqrt{ax + x\,\mathrm{d}ist + x.\mathrm{err}} &\leq (1+EPS/2)\sqrt{(ax \oplus (x\,\mathrm{d}ist \oplus x.\mathrm{err}))} \\ &\leq (1+2EPS) \otimes \sqrt[o]{(ax \oplus (x\,\mathrm{d}ist \oplus x.\mathrm{err}))}. \end{aligned}$$

□

Proposition 7.15. *If x is an AffApprox and $D = ax \ominus (1+EPS) \otimes (dist(x) \oplus x.err)$ is greater than zero, then $S(\sqrt{x}) \supseteq \sqrt{S(x)}$, where*

$$\sqrt{x} \equiv (r\_f.z; r\_f_0.z, r\_f_1.z, r\_f_2.z; r\_error)$$

with

$$\begin{aligned} r\_f &= \sqrt{x.f}, \\ t &= r\_f + r\_f, \\ r\_f_k &= AComplex(x.f_k.re, x.f_k.im; 0)/t. \end{aligned}$$

(Simply put, $r\_f_k = x.f_k/(2\sqrt{x.f})$. The reason we have to fuss to define r_f_k is because $\sqrt{x.f}$ is an AComplex.)

$$\begin{aligned} r\_error = (1+3EPS) & \\ \otimes \Big\{ & (1+EPS) \otimes \sqrt[o]{ax} \ominus (1-3EPS) \otimes [dist(x) \oslash (2 \times \sqrt[o]{ax}) \oplus \sqrt[o]{D}] \Big\} \\ \oplus & ((r\_f.e \oplus r\_f_0.e) \oplus (r\_f_1.e \oplus r\_f_2.e)). \end{aligned}$$

Proof. Let us work on the pieces.

$$\sqrt{ax} \leq (1 + EPS) \otimes \sqrt[o]{ax}.$$

Next,

$$\begin{aligned}
&\mathrm{d}ist(x)/(2\sqrt{ax}) + \sqrt{ax - (\mathrm{d}ist(x) + x.err)} \\
&\quad \geq (1 - EPS/2)^2 \mathrm{d}ist(x) \oslash (2\sqrt[o]{ax}) \\
&\qquad +(1 - EPS/2)^{1/2} \sqrt{ax \ominus (1 + EPS) \otimes (\mathrm{d}ist(x) \oplus x.err)} \\
&\quad \geq (1 - EPS/2)^3 \left[\mathrm{d}ist(x) \oslash (2\sqrt[o]{ax}) \oplus \sqrt[o]{D} \right] \\
&\quad \geq (1 + EPS/2)(1 - 3EPS) \left[\mathrm{d}ist(x) \oslash (2\sqrt[o]{ax}) \oplus \sqrt[o]{D} \right] \\
&\quad \geq (1 - 3EPS) \otimes \left[\mathrm{d}ist(x) \oslash (2\sqrt[o]{ax}) \oplus \sqrt[o]{D} \right].
\end{aligned}$$

Adding in the usual term, we get as our error bound

$$\begin{aligned}
&\sqrt{ax} - (\mathrm{d}ist(x)/(2\sqrt{ax}) + \sqrt{ax - (\mathrm{d}ist(x) + x.err)}\ ) \\
&\quad + ((r_f.e + r_f_0.e) + (r_f_1.e + r_f_2.e)) \\
&\leq (1 + EPS) \otimes \sqrt[o]{ax} - (1 - 3EPS) \otimes [\mathrm{d}ist(x) \oslash (2\sqrt[o]{ax}) \oplus \sqrt[o]{D}\ ] \\
&\quad + (1 + EPS/2)^2((r_f.e \oplus r_f_0.e) \oplus (r_f_1.e \oplus r_f_2.e)) \\
&\leq (1 + EPS/2)^3\big(\{(1 + EPS) \otimes \sqrt[o]{ax} \\
&\quad \ominus (1 - 3EPS) \otimes [\mathrm{d}ist(x) \oslash (2\sqrt[o]{ax}) \oplus \sqrt[o]{D}\ ]\} \\
&\quad \oplus ((r_f.e \oplus r_f_0.e) \oplus (r_f_1.e \oplus r_f_2.e))\big) \\
&\leq (1 + 3EPS) \otimes (\{(1 + EPS) \otimes \sqrt[o]{ax} \\
&\quad \ominus (1 - 3EPS) \otimes [\mathrm{d}ist(x) \oslash (2\sqrt[o]{ax}) \oplus \sqrt[o]{D}\ ]\} \\
&\quad \oplus((r_f.e \oplus r_f_0.e) \oplus (r_f_1.e \oplus r_f_2.e))).
\end{aligned}$$

□

Chapter 8

A trail of breadcrumbs about finding killer words

Here we outline our method for finding a decomposition of the initial box $\mathcal{W}$ into sub-boxes, each with a condition/killerword that kills off the entire associated sub-box, or with an indication that it belongs to one of the exceptional regions. For convenience, we will generally refer to a "sub-box" simply as a "box."

A naive approach uses diagonal enumeration to combine breadth-first enumeration of the tree of boxes with breadth-first enumeration of the tree of words (with three edges from each word, one for each generator or inverse-generator that isn't the inverse of the last symbol in the word, pointing to the concatenation of the word with the generator). The naive algorithm has running time $O(3^L 2^D)$, where L is maximum word length and D is maximum box depth. Much too slow - $3^{44}2^{120}$ operations is not close to reasonable.

To speed it up, there are three basic approaches, all of which are necessary:

1. we can avoid considering most boxes by stopping once we have a solution
2. we can reuse words that work on one box elsewhere
3. and we can use geometric heuristics to prefer words that are more likely to work.

The exposition will proceed in rough chronological order, in the hope that by describing some of the wrong turns, we'll help others avoid making the same mistakes.

The most obvious way of speeding up the search is to avoid the search entirely when feasible: a killerword works on a neighborhood of a region, and by testing killerwords found for nearby boxes, most of the time the search is not necessary.

Still, there are words of length as long as 44 that were considered, and testing all of the roughly 3^{44} combinations would be prohibitive. In practice (due to a bug), the search algorithm used for most of the parameter space was no better than the brute-force method just described, but to find killerwords for the remaining regions, an improvement was needed. Rather than blindly selecting words in first-in-first-out order, the algorithm can rank the

words under consideration based on a heuristic estimate of the likelihood of their being useful (a word is *u*seful if it is a prefix of a killerword). We note first that short words tend to be better than long words, as they have fewer steps and less error. Second, we note that words with a large translation distance are given a bad ranking, for two reasons: they will need more generators appended before they get back to the small translation distance which is needed for a contradiction, and computations with those words introduce more error per step than computations with closer words.

This approach was an improvement, but was not finding enough killerwords in the regions around X_3 and X_4. Further investigation showed that the algorithm was getting stuck on an identity: once it found an identity, it would consider only words which started with that identity, and ignored all of the other words. To fix this problem, a "diversity" heuristic was introduced, to give special consideration to unlikely but unusual words.

To prevent the search from running forever, it is temporarily abandoned after some number of steps, and re-done with twice as many steps every time the number of descendant boxes doubles. This way, the search could run forever, but only if the subdivision process runs forever. This merged process of alternately searching and subdividing we call the *d*ecomposition algorithm.

The decomposition algorithm went through several revisions; at each stage of the revision process, the algorithm effectively increased the extent to which killerwords found for one region were used to kill other regions. The first attempt—used to determine the feasibility of the whole effort— iterated over regions in depth-first order, performing the search as described above. At that stage, it became evident that the search process, as opposed to the subdivision process, was consuming nearly all of the computation time, and so the second version iterated over regions in breadth-first order, and, once it found a killerword, tried to use that word on all adjacent regions.

The breadth-first version was used to analyze the entire parameter space, although it skipped some parts due to various bugs; the search heuristic was replaced once, and there was considerable human input to tell the search about particularly difficult killerwords, or to tweak its search parameters (length, and weightings in the heuristic).

The third stage of the revision process reduced the number of boxes by attempting all found killerwords in a large region (about a thousand boxes) on all boxes in the region. It did not do any searching, since it was provided with a kist of killerwords known to work.

The final version was created when the bugs in evaluation were brought to light, and the existing killerword tree was found to be insufficient. It used the list of killerwords used for the entire tree, and some statistics about the number of subdivisions required in order for a given word to kill a particular box, and evaluated each word on each box. Whenever a word was evaluated, a kind of triage was used to determine whether that word was likely to kill the box in question, likely to kill any of its n^{th} generation descendants, or unlikely to kill any descendants of the box; the answer to that heuristic either allowed more detailed evaluation (with the error term included), deferred further evaluation until the box had been subdivided n more times, or excluded that word from further consideration on any descendant of the box. With these heuristics, this program wound up evaluating on average

about 10 of the roughly 13200 words per box, and was able to construct the tree consisting of the decomposition into sub-boxes with associated conditions/killerwords.

More recently, an updated version of the search program is in use to solve a nearby problem, classification by enumeration of cusped hyperbolic 3-manifolds. It's not far different from the final version of the search. Indeed, its source code was edited from the final version. The main differences are: the search for new words that work is again mixed with the search for words for all boxes; the search for words combines pairs of words instead of appending a generator; the logic for use of words is specific to the context of cusped manifolds. The updated code is available at https://github.com/njt99/momsearch.

Chapter 9

Computer code

Bibliography

[ACS] I. Agol, M. Culler, and P. Shalen, Dehn surgery, homology and hyperbolic volume, *Algebr. Geom. Topol.* **6** (2006), 2297–2312.

[ADST] I. Agol, P. Storm & W. Thurston (appendix by N. Dunfield), Lower bounds on volumes of hyperbolic Haken 3-manifolds, *J. AMS*, **20** (2007), 1053-1077.

[Bea] A. Beardon, *The Geometry of Discrete Groups*, Graduate Texts in mathematics **91**, Springer-Verlag,

[CLLMR] A. Champanerkar, J. Lewis, M. Lipyanskiy, S. Meltzer, (appendix by A. Reid) Exceptional regions and associated exceptional hyperbolic 3-manifolds, *Experiment. Math.* **16** (2007), no. 1, 107 - 118.

[F] W. Fenchel, Elementary Geometry in Hyperbolic Space, *de Gruyter Studies Math.*, **11**, de Gruyter, Berlin (1989).

[FS] L. H. de Figueiredo and J. Stolfi, Self-validated numerical methods and applications, *Brazilian Math. Colloq. Monograph*, IMPA, Rio de Janeiro, Brazil (1997).

[G] D. Gabai, On the Geometric and Topological Rigidity of Hyperbolic 3-Manifolds, to appear, *J. Amer. Math. Soc.* **10** (1997), no. 1, 37?74.

[G1] D. Gabai, The Smale conjecture for hyperbolic 3-manifolds: $\mathrm{Isom}(M^3) \cong \mathrm{Diff}(M^3)$, *J. Diff. Geom.* **58** (2001), 113-149.

[GMM] D. Gabai, R. Meyerhoff & P. Milley, Minimum volume cusped hyperbolic 3-manifolds, *J. AMS* **22** (2009), 1157-1215.

[GMT] D.Gabai, R. Meyerhoff & N. Thurston, Homotopy Hyperbolic 3-Manifolds are Hyperbolic, *Annals of Math.* (2) **157** (2003), 335–431.

[GT] D. Gabai & M. Trnkova, Exceptional hyperbolic 3-manifolds, *Comment. Math. Helv*, **90** (2015), 703-730.

[GHMTY] D. Gabai, R. Haraway, R. Meyerhoff, N. Thurston, A. Yarmola, Hyperbolic 3-manifolds of low cusp volume, preprint.

[GMTY] D. Gabai, R. Meyerhoff, N. Thurston, A. Yarmola, Enumerating Kleinian Groups, to appear, *Proceedings of Computational Aspects of Discrete Subgroups of Lie Groups*, AMS Contemporary Math. Series.

[IEEE] IEEE Standard for binary floating-point arithmetic (ANSI/IEEE Std 754-1985) published by the Institute of Electrical and Electronics Engineers, Inc., New York, NY, 1985.

[JR] K. Jones and A. Reid, Vol3 and other exceptional hyperbolic 3-manifolds, *Proc. A.M.S.* **129** (2001), 2175–2185.

[K1] W. Kahan, Interval arithmetic options in the proposed IEEE floating point arithmetic standard (Karl L. E. Nickel, ed.), in *Interval Mathematics*, 99–128 (1980).

[L] M. Lipyanskiy. A Computer-Assisted Application of Poincaré's Fundamental Polyhedron Theorem. Preprint, 2002.

[LM] M. Lackenby & R. Meyerhoff, The maximal number of exceptional Dehn surgeries, *Invent. Math.* **191** (2013), 341–382.

[M1] G. R. Meyerhoff, A Lower Bound for the Volume of Hyperbolic 3-Manifolds, *Canadian J. Math.* **39** (1987) 1038-1056.

[Mi] P. Milley, Minimum volume hyperbolic 3-manifolds, *J. Top.* **2**(2009), 181-192.

[Pe1] G. Perelman, Ricci flow with surgery on three-manifolds, arXiv:math.DG/0303109.

[Pe2] G. Perelman, The entropy formula for the Ricci flow and its geometric applications, arXiv:math.DG/0211159.