MPEJ

Mathematical Physics Electronic Journal

ISSN 1086-6655
Volume 1, 1995

Paper 6
Received: October 5, 1995, Revised: November 30, 1995, Accepted: December 6, 1995
Editor: J.-P. Eckmann

Bounds on the Zeros of a Renormalization Group Fixed Point

Hans Koch [1]
Department of Mathematics, University of Texas at Austin
Austin, TX 78712

Peter Wittwer [2]
Département de Physique Théorique, Université de Genève
Genève, CH 1211

Abstract. We prove that the Renormalization Group transformation for the Laplace transform of the $d = 3$ Dyson–Baker hierarchical model has a nontrivial entire analytic fixed point whose zeros all lie on the imaginary axis. Sharp upper and lower bounds on 80 of these zeros are used to verify the assumptions made in reference [11]. Our proof is computer–assisted.

[1] Supported in Part by the National Science Foundation under Grants DMS–9103590, DMS–9401422.
[2] Supported in Part by the Swiss National Science Foundation.

1. Introduction and Main Results

In this paper we present a computer–assisted proof of the following theorem. Let $\beta = 2^{-5/6}$, and consider the fixed point problem $\mathcal{N}(f) = f$ for the nonlinear transformation $\mathcal{N}$, defined by the equation

$$\bigl(\mathcal{N}(f)\bigr)(t^2) = \frac{1}{\sqrt{(1-\beta^2)\pi}} \int_{-\infty}^{\infty} ds\, e^{-\frac{1}{1-\beta^2}s^2} f\bigl((\beta t + s)^2\bigr)^2\,, \qquad t \in \mathbb{C}\,. \tag{1.1}$$

Theorem 1.1. *The transformation $\mathcal{N}$ has a fixed point f^* with the following properties.*
(a) f^* is an entire analytic function which takes real values when restricted to $\mathbb{R}$.
(b) The Taylor coefficients at zero of f^* are bounded in modulus by the corresponding coefficients of the function $z \mapsto K\exp\bigl(\frac{z}{10}\bigr)$, for some positive constant K.
(c) All zeros of f^* lie on the negative real axis.
In addition, f^* is of type Z, as defined below.

Definition 1.2. *A function f on $\mathbb{R}$ is of type Z if there are positive real numbers $x_0 < x_1 < \ldots < x_{79}$, such that*
(d) $f(-x_k) = 0$ for $k = 0, 1, \ldots, 79$.
(e) f has exactly 5 zeros in the interval $[-x_4, 0]$.
(f) $\sqrt{x_k} - \sqrt{x_{k-1}} < \frac{4}{3}$, for $k = 5, 6, \ldots, 79$.

Theorem 1.1, formulated in terms of the even function $t \mapsto f^*(t^2)$, was first announced in [11], where it was used to prove that for every $q < 3/5$, there are positive constants a and b such that

$$|f^*(-r)| \le a\exp\bigl(-br^q\bigr)\,, \qquad r \in \mathbb{R}_+\,. \tag{1.2}$$

This bound in turn implies that the function h^*, defined by the equation

$$h^*(t) = e^{-ct^2}\frac{c}{\pi}\int_{-\infty}^{\infty} ds\, e^{-2icst} f^*\bigl(-s^2\bigr)\,, \qquad t \in \mathbb{C}\,, \tag{1.3}$$

where $c = (2\alpha^2 - 1)/(4\alpha^2 - 1)$ and $\alpha = 2^{-1/6}$, is a nontrivial entire analytic fixed point of Baker's renormalization group transformation $\mathcal{R}$,

$$\bigl(\mathcal{R}(h)\bigr)(t) = 2\alpha \int_{-\infty}^{\infty} ds\, e^{-2cs^2} h(\alpha t + s) h(\alpha t - s)\,, \qquad t \in \mathbb{R}\,.$$

Interest in these fixed point problems stems from the theory of critical phenomena in statistical mechanics and quantum field theory: The third power of $\mathcal{R}$ is directly related to a renormalization group transformation on a space of scalar lattice field theories in three dimensions, with a certain hierarchical symmetry. We refer to [1,11] for further information on this aspect of the problem. Other rigorous results on scalar hierarchical models can be found in [2–12] and references therein.

In order to explain our strategy of proof, we shall now work towards the formulation of two lemmas which imply complementary parts of Theorem 1.1. The first of these lemmas covers all but part (*c*). Properties (*a*) and (*b*) hold for any function in the space $\mathcal{B}_{10}$, defined below, that will be used to prove the existence of the fixed point f^*.

Definition 1.3. *Let ρ be some fixed but arbitrary positive real number. Denote $\mathcal{B}_\rho$ the vector space of all entire analytic functions f, which take real values when restricted to $\mathbb{R}$, and which satisfy $\rho^n|f^{(n)}(0)| \to 0$ as n tends to infinity. Here, $f^{(n)}$ denotes the n–th derivative of f. When equipped with the following norm, $\mathcal{B}_\rho$ is a Banach space.*

$$\|f\|_\rho = \sup_n \rho^n \bigl|f^{(n)}(0)\bigr|\,, \qquad f \in \mathcal{B}_\rho\,. \tag{1.4}$$

The set of all functions in $\mathcal{B}_\rho$ whose derivatives at the origin are all nonnegative will be denoted by $\mathcal{B}_\rho^0$, and the set of all functions in $\mathcal{B}_\rho^0$ whose zeros all lie on the negative real axis will be denoted by $\mathcal{B}_\rho^+$. Finally, if f is an entire function and N a nonnegative integer, we define

$$\bigl(P_N f\bigr)(z) = \sum_{n=0}^{N} \frac{1}{n!} f^{(n)}(0)\, z^n\,, \qquad z \in \mathbb{C}. \tag{1.5}$$

Notice that $\mathcal{N}$ cannot have an attractive fixed point $f \not\equiv 0$, since $\mathcal{N}(f) = f$ implies that $D\mathcal{N}(f)f = 2f$. Here, $D\mathcal{N}(f)$ denotes the derivative of $\mathcal{N}$ at f. Numerically, the three largest eigenvalues of $D\mathcal{N}(f^*)$ are approximately 2, 1.427, and 0.859. In the context of computer–assisted proofs, the standard way of solving a hyperbolic fixed point problem $\mathcal{N}(f) = f$ is to convert it to a fixed point problem for a contraction $\mathcal{M}$. One of the canonical choices for $\mathcal{M}$ is a map of the form

$$\mathcal{M}(f) = f + M\bigl(\mathcal{N}(f) - f\bigr)\,, \tag{1.6}$$

where M is approximately the inverse of the operator $1 - D\mathcal{N}(f^*)$. In the case at hand, we define such a map $\mathcal{M}$ by picking an approximate fixed point f_0 for $\mathcal{N}$ and setting

$$M = \bigl[1 - P_\ell D\mathcal{N}(P_d f_0) P_\ell\bigr]^{-1}\,, \qquad \ell = 29,\ d = 100, \tag{1.7}$$

after verifying that the operator in square brackets is invertible.

Lemma 1.4. *Let $\rho = 10$. There exists a polynomial f_0 in $\mathcal{B}_\rho^0$, and a closed convex set V in $\mathcal{B}_\rho$ containing f_0, such that the following holds. The transformation $\mathcal{M}$ defined above is of class C^∞ on $\mathcal{B}_\rho$ and maps V to itself. The operator norm of $D\mathcal{M}(f)$ is less than 1, for all f in V. Furthermore, every function f in $\mathcal{M}(V)$ is of type Z; see Definition 1.2.*

This lemma implies that $\mathcal{M}$ is a contraction on the set V. Thus, by the contraction mapping principle, $\mathcal{M}$ has a unique fixed point f^* in V. In addition, f^* has all the properties described in Theorem 1.1, with the possible exception of (*c*).

Our proof of Lemma 1.4 is computer–assisted. By following a strategy that has been used successfully with several other fixed point problems, we derive sharp bounds on a certain number of Taylor coefficients for f^*, and on the norm of the remaining higher order part. The novel aspects of our proof stem mainly from the need for high accuracy: The values of the function f^* need to be estimated at a wide range of points, and on the negative real axis where the terms of the Taylor series are alternating in sign (the maximal value of $|f^*(x)|/f^*(|x|)$ on the interval between the last two zeros considered is less than 10^{-150}). In order to give an idea of what accuracy is used to meet all requirements, let us mention that the approximate fixed point f_0 satisfies the bound $\|\mathcal{M}(f_0) - f_0\|_{10} < 10^{-365}$, and that the set V in Lemma 1.4 specifies the first 1001 Taylor coefficients of every function $f \in V$ to a precision of up to 370 decimal digits.

We note that a contraction $\mathcal{M}$ of the type considered here is ideally suited for high precision bounds: The map $h \mapsto \mathcal{M}(f_0 + h) - f_0$ can be implemented in a nearly cancellation–free form since it decomposes easily into an affine part and a remainder that is explicitly of higher order in h. In fact, other "standard" choices would have made our computer–assisted proof prohibitively time consuming. The only problem with $\mathcal{M}$ is that it lacks the following important property of the transformation $\mathcal{N}$: If a function $f \in \mathcal{B}_\rho$ has no zeros off the negative real axis, then the same is true for $\mathcal{N}(f)$. Here, we have to assume that ρ larger than a certain number $\hat{\rho} < 4$. This property is an immediate consequence of Corollary 2.1 in [11]. Further details will be given in Section 3.

Let us now turn to the proof of part (c) of Theorem 1.1. Assume that $\rho > \hat{\rho}$. Our strategy is to consider a new contraction $\mathcal{K}$ which has f^* as a fixed point, and which leaves the set $\mathcal{B}_\rho^+ \cap W$ invariant for some given neighborhood W of f^*. One of the tasks will be to show that the intersection of $\mathcal{B}_\rho^+$ with W is non–empty. This is done by constructing an approximate fixed point $f_2 \in W$ whose zeros all lie on the negative real axis.

The map $\mathcal{K}$ will be chosen of the form $\mathcal{N} \circ \mathcal{U}^*$, where $\mathcal{U}^*$ is a suitable (nonlinear) projection that leaves f^* invariant. Such a map is a contraction near f^* if $D\mathcal{N}(f^*)$ contracts vectors in the range of $D\mathcal{U}^*(f^*)$.

Given a vector $u = (u_0, u_1, u_2, u_3)$ in $\mathbb{R}^4$ and a function f in $\mathcal{B}_\rho$, let $\mathcal{U}(u, f)$ be the function defined by the equation

$$\bigl(\mathcal{U}(u, f)\bigr)(z) = u_0 e^{u_1 z} f(u_2 z + u_3)\,. \tag{1.8}$$

In Section 3 we will specify real numbers $v > 0$ and $\hat{\rho} < r < \rho$, such that $\mathcal{U}(u, f)$ lies in $\mathcal{B}_r$ whenever u is an element of $U = \{u \in \mathbb{R}^4\colon |u_1| < v, 0 < u_2 < 1 + v\}$ and $f \in \mathcal{B}_\rho$.

Definition 1.5. *Let f be a function in $\mathcal{B}_\rho$, and denote by U_f the set of all vectors $u = (u_0, u_1, u_2, u_3)$ in U such that f has no zero in $[u_3, 0]$, if $u_3 \le 0$. Then we define $\mathcal{K}(f) = \mathcal{N}\bigl(\mathcal{U}(u^*(f), f)\bigr)$, where $u^*(f)$ is the vector closest to $(1, 0, 1, 0)$ among all vectors u in U_f that satisfy the equation*

$$P_3\mathcal{U}(u, f) = P_3 f^*\,. \tag{1.9}$$

If there is no such vector $u^*(f)$, or if it is not unique, then $\mathcal{K}(f)$ remains undefined.

Notice that U_{f^*} is non–empty, since $f^*(0) \neq 0$. Thus, $u^*(f^*) = (1,0,1,0)$, so that f^* is a fixed point of the transformation $\mathcal{K}$. Our reason for introducing the sets U_f is the following. Consider a function $f \in \mathcal{B}_\rho^+$. Then the condition $u \in U_f$ ensures that $\mathcal{U}(u,f)$ has no zeros off the negative real axis. Recalling the abovementioned property of $\mathcal{N}$, we conclude that the function $\mathcal{K}(f)$, if defined, lies again in $\mathcal{B}_\rho^+$.

In what follows, if L is a continuous linear operator on $\mathcal{B}_\rho$, then $\|L\|_\rho$ will denote the operator norm of L.

Lemma 1.6. *Let $\rho = 6$ and $n = 28$. There are positive constants γ and δ, an invertible linear operator Γ on $\mathcal{B}_\rho$ satisfying $\Gamma - \gamma = P_n(\Gamma - \gamma)P_n$, and a function $f_2 \in \mathcal{B}_\rho^+$, such that the following holds. The transformation $\mathcal{K}$ is well defined and of class C^∞ on (a neighborhood of) a closed convex set W that contains both $\{f \in \mathcal{B}_\rho : \|\Gamma^{-1}[f - f_2]\|_\rho \leq 5\delta\}$ and the set $\mathcal{M}(V)$ described in Lemma 1.4. Furthermore, the function f_2 satisfies the bound $\|\Gamma^{-1}[\mathcal{K}(f_2) - f_2]\|_\rho \leq \delta$, and the derivative of $\mathcal{K}$ satisfies the bound $\|\Gamma^{-1}D\mathcal{K}(f)\Gamma\|_\rho < \frac{4}{5}$, for all f in W.*

By the contraction mapping principle, this lemma implies that $\mathcal{K}$ has a unique fixed point in W. Given that W contains the set $\mathcal{M}(V)$, this fixed point has to be the same as the fixed point f^* of $\mathcal{M}$ and $\mathcal{N}$. Thus, since $W \cap \mathcal{B}_6^+ \ni f_2$ is a non–empty closed subset of W that is invariant under $\mathcal{K}$, it follows that f^* lies in $\mathcal{B}_6^+$.

Remarks.

- The analysis presented here uses input data which are the results of prior numerical investigations. Among those input data are the polynomial f_0, the set V, and 5000 approximate zeros for f^* (of which the first 80 are shown to be accurate to within $\pm 10^{-40}$). Most of these zeros were obtained by iterating a contraction associated with a version of $\mathcal{N}$ that acts directly on zeros. This iteration was started with 4 zeros.
- Unlike the polynomial f_0, the approximate fixed point f_2 for $\mathcal{K}$ is not given beforehand. It is determined, as part of the proof of Lemma 1.6, from a polynomial f_1 which is the canonical product associated with the abovementioned approximate zeros for f^*. Despite its high degree, the polynomial f_1 is too far away from f^* to be useful directly. A better approximation f_2 is obtained from f_1 by applying a sequence of 80 transformations of the form $f \mapsto \mathcal{N}\big(\mathcal{U}(u,f)\big)$, where u is chosen appropriately at each step. This method is an alternative to fine–tuning the coefficients of f_1 such as to get a polynomial very close to the stable manifold of $\mathcal{N}$ at f^*, and then iterating the transformation $\mathcal{N}$ a large number of times.
- Our proof of the preceding two lemmas incorporates several features that are not commonly found in computer–assisted proofs. This includes the use of high precision floating point arithmetic. On a higher level, precision was increased by eliminating the need for a "general error" term in our standard sets (the analogue of intervals) for analytic functions. Related to this is the fact that we use a weighted sup–norm for Taylor coefficients, as opposed to the usual ℓ^1–type norms. Some advantages of these choices will be pointed out in subsequent sections. As a general rule, we have tried to

use methods that are both efficient and conceptually simple; see e.g. the proof for the existence and local uniqueness of the fixed point f^* for $\mathcal{M}$.

The remaining part of this paper is organized as follows. Section 2 contains the estimates that are used (a) in our programs to bound the results of elementary operations on the spaces $\mathcal{B}_\rho$, and (b) in Section 3 to establish some basic properties of the contractions $\mathcal{M}$ and $\mathcal{N}$. In Section 4 we will describe the main steps in the proof of Lemma 1.4 and Lemma 1.6, with reference to the corresponding computer programs. A more detailed discussion of these programs is given in Section 5. For an exact description of our proof we refer to the source code (the file `Section.6` in the supplement to this MPEJ paper).

2. Some Operations Involving Analytic Functions

In order to estimate expressions involving the transformations $\mathcal{M}$ or $\mathcal{K}$, it is necessary to first consider some elementary maps that enter the definition of these transformations.

Most of the bounds given here are close to optimal and very easy to prove, due to the following fact which is related to our choice of norms. Consider the vector space $\mathcal{P}$ of all polynomials over $\mathbb{C}$ with real coefficients, and denote by ϵ_ρ^n the Taylor polynomial of degree n for the function $z \mapsto \exp(z/\rho)$,

$$\epsilon_\rho^n(z) = \sum_{k=0}^{n} \frac{1}{\rho^k k!} z^k , \qquad z \in \mathbb{C}. \tag{2.1}$$

Proposition 2.1. *Let $\rho_0, \rho_1, \ldots, \rho_k$ be positive real numbers, and let T_0 be a k–linear map from $\mathcal{P}^k$ to $\mathcal{P}$. Assume that the Taylor coefficients of $T_0(f_1, f_2, \ldots, f_k)$ are linear combinations with positive coefficients of products of Taylor coefficients of the functions $f_1, f_2, \ldots, f_k$, and that there exists a constant K such that $\|T_0(\epsilon_{\rho_1}^{n_1}, \epsilon_{\rho_2}^{n_2}, \ldots, \epsilon_{\rho_k}^{n_k})\|_{\rho_0} \le K$ for all $n_1, n_2, \ldots, n_k$. Then T_0 can be extended in a unique way to a continuous k–linear operator T from $\mathcal{B}_{\rho_1} \times \mathcal{B}_{\rho_2} \times \ldots \times \mathcal{B}_{\rho_k}$ to $\mathcal{B}_{\rho_0}$, and this extension satisfies*

$$\|T(f_1, f_2, \ldots, f_k)\|_{\rho_0} \le K \|f_1\|_{\rho_1} \|f_2\|_{\rho_2} \cdots \|f_k\|_{\rho_k} , \qquad f_i \in \mathcal{B}_{\rho_i} , \ 1 \le i \le k .$$

The proof of this proposition is straightforward and will be omitted.

In what follows, the symbol $c_n(f)$ will be used to denote the n^{th} Taylor coefficient of a function f at zero. In addition, N is assumed to be some fixed positive integer; and for every $f \in \mathcal{B}_\rho$ we define

$$e_\rho(f) = \frac{1}{\rho^{N+1}(N+1)!} \|(1 - P_N) f\|_\rho . \tag{2.2}$$

2.1. Translation and Evaluation

Consider the translation operators $T_\lambda : f \mapsto f(.-\lambda)$ for real values of λ.

Proposition 2.2. *Let $\rho > 0$ and $\lambda \in \mathbb{R}$ be given. Then T_λ maps $\mathcal{B}_\rho$ into itself and satisfies $\|T_\lambda f\|_\rho \le \exp(|\lambda|/\rho)\|f\|_\rho$ for every $f \in \mathcal{B}_\rho$. If f is a function in $\mathcal{B}_\rho$ such that $P_N f = 0$, and if $0 \le n \le N$, then*

$$\bigl|c_n(T_\lambda f)\bigr| \le |\lambda|^{N+1-n}\binom{N+1}{n} e^{|\lambda|/\rho} e_\rho(f)\,. \tag{2.3}$$

Under the additional assumption that $|\lambda|/\rho < N+2-n$, we also have the bound

$$\bigl|c_n(T_\lambda f)\bigr| \le |\lambda|^{N+1-n}\binom{N+1}{n}\left(1-\frac{|\lambda|/\rho}{N+2-n}\right)^{-1} e_\rho(f)\,. \tag{2.4}$$

Proof. If λ is nonnegative, then the fact that T_λ maps $\mathcal{B}_\rho$ into itself, and the given bound on $\|T_\lambda f\|_\rho$, follow immediately from Proposition 2.1. But since reflection $f \mapsto f(-.)$ is norm–preserving on $\mathcal{B}_\rho$, the same holds for any $\lambda \in \mathbb{R}$.

In order to prove the bound (2.3), assume now that $P_N f = 0$. By using that $|c_m(f)|$ is less than or equal to $\|f\|_\rho/(\rho^m m!)$, we obtain

$$\begin{aligned}\bigl|c_n(T_\lambda f)\bigr| &= \left|\sum_{m=N+1}^{\infty} c_m(f)\binom{m}{n}\lambda^{m-n}\right| \le \|f\|_\rho \frac{1}{\rho^n n!}\sum_{m=N+1}^{\infty}\frac{1}{(m-n)!}\left(\frac{|\lambda|}{\rho}\right)^{m-n}\\ &= e_\rho(f)\,\rho^{N+1-n}\frac{(N+1)!}{n!}\sum_{k=N+1-n}^{\infty}\frac{1}{k!}\left(\frac{|\lambda|}{\rho}\right)^{k}.\end{aligned} \tag{2.5}$$

By Taylor's theorem, the last sum can be bounded by its first term times $\exp(|\lambda|/\rho)$. The result is precisely the inequality (2.3). In addition, if $r = |\lambda|\rho^{-1}/(N+2-n)$ is less than one, then the same sum can be bounded by a geometric series with ratio r. This proves inequality (2.4).

■

By using the bound (2.4) for $n = 0$, and the fact that $f(x) = c_0(T_{-x}f)$, we immediately obtain the following corollary.

Corollary 2.3. *Let $\rho > 0$ and $x \in \mathbb{R}$ be given. Then the evaluation functional $f \mapsto f(x)$ is continuous on $\mathcal{B}_\rho$. If $|x|/\rho < N+2$, then the following holds for all functions $f \in \mathcal{B}_\rho$ that satisfy $P_N f = 0$.*

$$|f(x)| \le |x|^{N+1}\left(1-\frac{|x|/\rho}{N+2}\right)^{-1} e_\rho(f)\,. \tag{2.6}$$

2.2. The Product of Functions

Proposition 2.4. *Let $\sigma, \tau > 0$ be given, and define $\rho = \frac{\sigma\tau}{\sigma+\tau}$. If f and g are functions in $\mathcal{B}_\sigma$ and $\mathcal{B}_\tau$, respectively, then the product fg is an element of $\mathcal{B}_\rho$, and*

$$\|fg\|_\rho \leq \|f\|_\sigma \|g\|_\tau \,. \tag{2.7}$$

Under the additional assumption that $P_N g = 0$, we also have the bound

$$e_\rho(fg) \leq (\tau/\rho)^{N+1} \|f\|_\sigma \, e_\tau(g) \,. \tag{2.8}$$

Proof. The bound (2.7) is obtained by using Proposition 2.1, and (2.8) follows since $P_N g = 0$ implies that $P_N(fg) = 0$.

$\blacksquare$

Proposition 2.5. *Let $\sigma > 0$ and $f \in \mathcal{B}_\sigma$ be given, and let $\rho = \sigma(N+1)/(N+2)$. Then the function Zf, defined by the equation $(Zf)(z) = zf(z)$, is in $\mathcal{B}_\rho$. Furthermore, if $P_N f = 0$, then*

$$e_\rho(Zf) \leq (N+1)\sigma \, e_\sigma(f) \,. \tag{2.9}$$

Proof. Consider the function $\varphi : x \mapsto x(\rho/\sigma)^x$ on $\mathbb{R}_+$. Since $\ln\varphi$ has a strictly negative second derivative, and since $\varphi(N+2) = \varphi(N+1)$, the restriction to $\mathbb{N}$ of φ takes its maximum at $N+1$ and $N+2$. Thus, it follows that

$$\begin{aligned} e_\rho(Zf) &= \left[\rho^{N+1}(N+1)!\right]^{-1} \sup_{n \geq N+2} \rho^n n! \, |c_{n-1}(f)| \\ &\leq \left[\rho^{N+1}(N+1)!\right]^{-1} \sup_{n \geq N+2} (\rho/\sigma)^n n\sigma \|f\|_\sigma = (N+1)\sigma \, e_\sigma(f) \,. \end{aligned} \tag{2.10}$$

$\blacksquare$

2.3. The Derivative and a Difference Quotient

The proof of the following proposition is straightforward.

Proposition 2.6. *Let $\rho > 0$ and $f \in \mathcal{B}_\rho$ be given. Denote by Df and Δf the first derivative of f and the function $z \mapsto [f(z) - f(0)]/z$, respectively. Then Df and Δf are in $\mathcal{B}_\rho$, and*

$$\begin{aligned} |c_N(Df)| &\leq (N+1) \, e_\rho(f) \,, & e_\rho(Df) &\leq \tfrac{1}{\rho} \, e_\rho(f) \,, \\ |c_N(\Delta f)| &\leq e_\rho(f) \,, & e_\rho(\Delta f) &\leq \tfrac{1}{\rho(N+2)} \, e_\rho(f) \,. \end{aligned} \tag{2.11}$$

2.4. Convolution with Gaussians

Throughout this subsection, λ will be a fixed but arbitrary positive real number. If f is a function in $\mathcal{B}_\sigma$ for some $\sigma > 4\lambda$, define a function $H_\lambda f$ by the equation

$$\bigl(H_\lambda f\bigr)\bigl(t^2\bigr) = \frac{1}{\sqrt{4\pi\lambda}} \int_{-\infty}^{\infty} ds\, e^{-\frac{1}{4\lambda}s^2} f\bigl((t-s)^2\bigr)\,, \qquad t \in \mathbb{C}\,. \tag{2.12}$$

By applying H_λ to the function $z \mapsto \exp(z/\sigma)$, we obtain immediately the following fact.

Proposition 2.7. *Let $\rho > 0$ be given, and define $\sigma = \rho + 4\lambda$. Then H_λ is a bounded linear operator from $\mathcal{B}_\sigma$ to $\mathcal{B}_\rho$, and*

$$\|H_\lambda f\|_\rho \le \sqrt{\sigma/\rho}\, \|f\|_\sigma\,, \qquad f \in \mathcal{B}_\sigma\,. \tag{2.13}$$

A straightforward calculation shows that the Taylor coefficients of $H_\lambda f$ and f are related by the equation

$$\rho^n n!\, c_n(H_\lambda f) = \sum_{k=n}^{\infty} J_{nk}\, \sigma^k k!\, c_k(f)\,, \tag{2.14}$$

where

$$J_{nk} = \frac{(2k)!}{(2n)!(k-n)!}\, \frac{\rho^n n!}{\sigma^k k!}\, \lambda^{k-n}\,. \tag{2.15}$$

Proposition 2.8. *Let $\rho > 0$ and $\sigma = \rho + 4\lambda$. If f is a function in $\mathcal{B}_\sigma$ such that $P_N f = 0$, and if $0 \le n \le N$, then*

$$\bigl|c_n\bigl(H_\lambda f\bigr)\bigr| \le \left[\frac{1}{\rho^n n!}\sqrt{\sigma/\rho} - c_n\bigl(H_\lambda \epsilon_\sigma^N\bigr)\right] \sigma^{N+1}(N+1)!\, e_\sigma(f)\,. \tag{2.16}$$

Proof. By linearity, we may assume that $\|f\|_\sigma = 1$. Then the right hand side of (2.16) is equal to $c_n(H_\lambda g)$, where g is the function defined by the equation

$$g(x) = e^{x/\sigma} - \epsilon_\sigma^N(x) = \sum_{k=N+1}^{\infty} \frac{1}{\sigma^k k!}\, x^k\,, \qquad x \in \mathbb{C}. \tag{2.17}$$

Thus, since the Taylor coefficients of f are bounded in absolute value by the corresponding coefficients of g, and since $J_{nk} \ge 0$ for all $k \ge n \ge 0$, the bound (2.16) follows.

■

We note that the bound (2.16) is optimal. But in order to be useful, it requires an estimate on the difference in the square bracket. In practice, the bound on this difference deteriorates for small n since the two terms are almost equal in size (if N is large). The following proposition provides an alternative in this case.

Proposition 2.9. *Let $\rho > 0$ and $\sigma = \rho + 4\lambda$. Define $r_0 = 4\lambda/\sigma$ and*

$$r_n = \frac{4\lambda}{\sigma}\,\frac{N+3/2}{N+2-n}\,, \qquad n = 1,2,\ldots,N\,. \tag{2.18}$$

If f is a function in $\mathcal{B}_\sigma$ and $0 \le n \le N$, such that $P_N f = 0$ and $r_n < 1$, then

$$\bigl|c_n\bigl(H_\lambda f\bigr)\bigr| \le \frac{1}{1-r_n}\,\frac{(2N+2)!}{(2n)!(N+1-n)!}\,\lambda^{N+1-n}\,e_\sigma(f)\,. \tag{2.19}$$

Proof. Assume that ρ, f, and n satisfy the given hypotheses. Then for all $k > N$, we have

$$\frac{J_{n,k+1}}{J_{nk}} = \frac{4\lambda}{\sigma}\,\frac{k+1/2}{k+1-n} \le r_n \tag{2.20}$$

and $\sigma^k k!\,|c_k(f)| \le \|f\|_\sigma$. Thus, the n^{th} Taylor coefficient of $H_\lambda f$ satisfies the bound

$$\bigl|c_n\bigl(H_\lambda f\bigr)\bigr| \le \frac{1}{\rho^n n!}\sum_{k=N+1}^{\infty} J_{nk}\,\|f\|_\sigma \le \frac{1}{\rho^n n!}\,\frac{1}{1-r_n} J_{n,N+1}\,\|f\|_\sigma\,. \tag{2.21}$$

The expression after the second inequality in (2.21) is precisely the right hand side of (2.19).

■

3. The Contractions $\mathcal{M}$ and $\mathcal{K}$

In this section we will consider the domains and differentiability properties of the transformations $\mathcal{M}$ and $\mathcal{K}$. Explicit expressions will be given for the first derivatives of these two transformations.

In what follows, ρ is assumed to be a fixed but arbitrary real number satisfying

$$\rho > \frac{2-2\beta^2}{1-2\beta^2} \quad (= 3.702\ldots)\,. \tag{3.1}$$

In particular, this inequality holds for the values of ρ that are used in Lemma 1.4 and Lemma 1.6. Given such a choice of ρ, set

$$\lambda = \frac{1-\beta^2}{4\beta^2}\,, \qquad \sigma = \rho + 4\lambda\,, \qquad \tau = 2\sigma\,. \tag{3.2}$$

Define $Q(f) = f^2$ and $S(f) = f(\beta^2.)$. Then the transformation $\mathcal{N}$, given by equation (1.1), can be written as a composition $\mathcal{N} = H_\lambda \circ Q \circ S$, where H_λ is the convolution operator defined in (2.12).

Proposition 3.1. *If $r \ge \beta^2\tau$ then $\mathcal{N}$ is a C^∞ map from $\mathcal{B}_r$ to $\mathcal{B}_\rho$.*

Proof. By Proposition 2.7 and Proposition 2.4, the maps H_λ and Q are of class C^∞ from $\mathcal{B}_\tau$ to $\mathcal{B}_\sigma$ and from $\mathcal{B}_\sigma$ to $\mathcal{B}_\rho$, respectively. Since S is clearly bounded from $\mathcal{B}_r$ to $\mathcal{B}_\tau$, the assertion follows.

■

We note that the condition (3.1) on ρ ensures that the hypothesis of Proposition 3.1 is satisfied for the particular choice $r = \rho$. The derivative of $\mathcal{N}$ is given by the equation

$$\big(D\mathcal{N}(f)h\big)(t^2) = \frac{2}{\sqrt{(1-\beta^2)\pi}} \int_{-\infty}^{\infty} ds\, e^{-\frac{1}{1-\beta^2}s^2} f\big((\beta t+s)^2\big) h\big((\beta t+s)^2\big)\,, \quad t \in \mathbb{C}\,. \tag{3.3}$$

Consider now the transformation $\mathcal{M}$ given by equation (1.6), where M is the inverse (if it exists) on $\mathcal{B}_\rho$ of the operator

$$M' = 1 - P_\ell D\mathcal{N}(P_d f_0) P_\ell\,, \tag{3.4}$$

associated with some approximate fixed point f_0 of $\mathcal{N}$ and two positive integers ℓ and d (which will eventually be set to 29 and 100, respectively). The following is an immediate consequence of Proposition 3.1, given that the condition (3.1) implies $\rho > \beta^2\tau$.

Corollary 3.2. *If M' is invertible on $P_\ell\mathcal{B}_\rho$, then $\mathcal{M}$ is a C^∞ map on $\mathcal{B}_\rho$.*

Our reason for considering the transformation $\mathcal{M}$, in place of $\mathcal{N}$, is that it can be expected to be a contraction near f_0, if ℓ and d are sufficiently large. The cancellations that occur when passing from $\mathcal{N}$ to $\mathcal{M}$ can be seen explicitly in the following decomposition of $D\mathcal{M}$.

$$\begin{aligned} D\mathcal{M}(f) &= M\big[(1-P_\ell)D\mathcal{N}(f) + P_\ell D\mathcal{N}(f)(1-P_\ell) + P_\ell D\mathcal{N}(f)P_\ell - P_\ell - (1-P_\ell)\big] + 1 \\ &= (1-P_\ell)D\mathcal{N}(f) + M\big[P_\ell D\mathcal{N}(f)(1-P_\ell) + P_\ell\big(D\mathcal{N}(f) - D\mathcal{N}(P_d f_0)\big)P_\ell \\ &\qquad + P_\ell D\mathcal{N}(P_d f_0)P_\ell - P_\ell\big] + P_\ell \\ &= (1-P_\ell)D\mathcal{N}(f) + MP_\ell D\mathcal{N}(f)(1-P_\ell) + MP_\ell D\mathcal{N}(f - P_d f_0)P_\ell\,. \end{aligned} \tag{3.5}$$

This representation will be used to show that under the hypothesis of Lemma 1.4, the derivative $D\mathcal{M}(f)$ maps the unit ball of $\mathcal{B}_\rho$ into a ball of radius strictly smaller than 1, for every function f in a given neighborhood V of f_0. A more detailed description will be given in Section 4.

Next, we shall consider the transformation $\mathcal{K}$ defined in Section 1, assuming the existence of a fixed point f^* for $\mathcal{N}$, as described in Lemma 1.4. The definition of $\mathcal{K}$ involves the map $\mathcal{U}$, given by equation (1.8), from $\mathbb{R}^4 \times \mathcal{B}_\rho$ to the vector space of entire analytic functions. In order to restrict the range of $\mathcal{U}$ appropriately, only vectors u in a certain subset U of $\mathbb{R}^4$ will be considered. The set U is of the form

$$U = \big\{u \in \mathbb{R}^4 \colon |u_1| < v, 0 < u_2 < 1 + v\big\}\,, \quad v = \left(\frac{\rho}{\beta^2\tau} - 1\right)\left(1 + 4\rho\frac{N+2}{N+1}\right)^{-1}, \tag{3.6}$$

where N is some fixed positive integer ($N = 199$ in the proof of Lemma 1.6, and $\rho = 6$).

Proposition 3.3. *If $r = \beta^2\tau$, then $\mathcal{U}$ maps $U \times \mathcal{B}_\rho$ to $\mathcal{B}_r$ and is of class* C^∞.

Proof. Let (u, f) be a fixed element of $U \times \mathcal{B}_\rho$, and let $b_1 < v$ and $b_2 < 1+v$ be positive upper bounds on $|u_1|$ and $|u_2|$, respectively. For $n = 0, 1, \ldots$, define

$$\xi_{u,n}(z) = u_0 z^n \exp(u_1 z)\,, \qquad \hat{f}_{u,n}(z) = f^{(n)}(u_2 z + u_3)\,. \tag{3.7}$$

By using the results of Section 2, it is easy to see that the functions $\xi_{u,n}$ and $\hat{f}_{u,n}$ lie in $\mathcal{B}_s$ and $\mathcal{B}_t$, respectively, whenever $s|u_1| < 1$ and $t|u_2| \le \rho$. The particular values of s and t that will be used later and that clearly have the required properties, are

$$s = \frac{1}{4b_1}\frac{N+1}{N+2}\,, \qquad t = \frac{\rho}{b_2}\,. \tag{3.8}$$

A short calculation shows that $r < st/(s+t)$. Thus, by Proposition 2.4, the functions $\xi_{u,m}\hat{f}_{u,n}$ are all elements of $\mathcal{B}_r$. Consider now the map $\varphi\colon u \mapsto \mathcal{U}(u, f)$ from U to $\mathcal{B}_r$, with f fixed as above. Since the partial derivatives of φ at u are linear combinations of functions $\xi_{u,m}\hat{f}_{u,n}$, with $m+n$ ranging from zero to twice the order of the derivative, it follows that φ is of class C^∞. But since $\mathcal{U}$ is linear in its second argument, this suffices to conclude that $\mathcal{U}$ is C^∞ as well.

■

We note that the assumption (choice of v) in Proposition 3.3 is more restrictive than would be necessary at this point, but it allows us to give a proof (choice of parameters) that can be applied later to get accurate bounds for the map $\mathcal{K}$.

The corollary below will be used to check whether a given open neighborhood W' of the fixed point f^* of $\mathcal{N}$ is contained in the domain of $\mathcal{K}$. In particular, for every $f \in W'$, we have to be able to solve the equation $\mathcal{F}(u, f) = 0$, where

$$\mathcal{F}(u, f) = P_3\mathcal{U}(u, f) - P_3 f^*\,. \tag{3.9}$$

This will be done by using the associated Newton map $u \mapsto u - [D_1\mathcal{F}(u, f)]^{-1}\mathcal{F}(u, f)$. Here, and in what follows, we use the symbol D_n to denote the partial derivative with respect to the n^{th} argument. In addition, $\mathcal{F}$ will be regarded as a map to the space $P_3\mathcal{B}_r$, which may be thought of as $\mathbb{R}^4$ by identifying a polynomial $z \mapsto u_0 + u_1 z + u_2 z^2 + u_3 z^3$ with the vector of its coefficients (u_0, u_1, u_2, u_3). On $\mathbb{R}^4$, we consider the following family of norms

$$\|u\|_\omega = \sup\{\omega_i^{-1}|u_i| : i = 0, 1, 2, 3\}\,, \qquad u \in \mathbb{R}^4, \tag{3.10}$$

associated with vectors ω in $\mathbb{R}^4_+$.

Corollary 3.4. *Assume that there exists a vector $\omega \in \mathbb{R}^4_+$, a closed ball $B \subset U$ containing the point $(1, 0, 1, 0)$, and an open neighborhood W' of f^* in $\mathcal{B}_\rho$, such that* *for every $f \in W'$ the following holds: The set U_f contains B, and the Newton map for the equation $\mathcal{F}(u,f)=0$ is a contraction on B. Then $\mathcal{K}$ is well defined and of class* C^∞ *as a map from W' to $\mathcal{B}_\rho$.*

Proof. Assume that the given hypotheses are satisfied, and let $r=\beta^2\tau$. Then for every f in W', the equation $\mathcal{F}(u,f)=0$ has a unique solution in B. Following Definition 1.5 (with the appropriate interpretation of "closest"), this solution will be denoted by $u^*(f)$, and we define $\mathcal{K}(f)=\mathcal{N}\bigl(\mathcal{U}(u^*(f),f)\bigr)$. Since by Proposition 3.3, $\mathcal{F}$ is a C^∞ map from $U\times\mathcal{B}_\rho$ to $P_3\mathcal{B}_r$, the implicit function theorem can be used to conclude that the map $f\mapsto u^*(f)$ is C^∞ on W'. The assertion now follows from Proposition 3.1 and Proposition 3.3.

■

Let us now assume that the hypotheses of Corollary 3.4 are satisfied. In order to prove that $\mathcal{K}$ is a contraction on some given set $W\subset W'$, we first need to find an explicit expression for the derivative of the map $\mathcal{U}^*: f\mapsto\mathcal{U}(u^*(f),f)$. In what follows, u and f will always denote elements of U and W, respectively. By using the functions defined in (3.7), the first partial derivatives of $\mathcal{U}$ can be written as

$$\begin{aligned} D_1\mathcal{U}(u,f)w &= w_0\xi_{u,0}\hat{f}_{u,0}+w_1u_0\xi_{u,1}\hat{f}_{u,0}+w_2u_0\xi_{u,1}\hat{f}_{u,1}+w_3u_0\xi_{u,0}\hat{f}_{u,1}\,,\\ D_2\mathcal{U}(u,f)h &= \mathcal{U}(u,h)=u_0\xi_{u,0}\hat{h}_{u,0}\,, \end{aligned}\tag{3.11}$$

and the corresponding derivatives of $\mathcal{F}$ and are obtained by applying the projection P_3 to the expressions on the right hand side of (3.11). For the derivative of u^* we get

$$Du^*(f)=-\bigl[D_1\mathcal{F}\bigl(u^*(f),f\bigr)\bigr]^{-1}D_2\mathcal{F}\bigl(u^*(f),f\bigr)\,,\tag{3.12}$$

as a consequence of the identity $\mathcal{F}(u^*(.),.)\equiv 0$. Let $r=\beta^2\tau$. Then the derivative of $\mathcal{U}^*$ can be written as follows.

$$\begin{aligned} D\mathcal{U}^*(f) &= D_2\mathcal{U}\bigl(u^*(f),f\bigr)+D_1\mathcal{U}\bigl(u^*(f),f\bigr)Du^*(f)\\ &= (1-P_3)\bigl[1-\mathcal{V}(f)P_3\bigr]D_2\mathcal{U}\bigl(u^*(f),f\bigr)\,, \end{aligned}\tag{3.13}$$

where $\mathcal{V}(f)$ is the linear operator from $P_3\mathcal{B}_r$ to $\mathcal{B}_r$, defined by the equation

$$\mathcal{V}(f)=D_1\mathcal{U}\bigl(u^*(f),f)\bigr)\bigl[D_1\mathcal{F}\bigl(u^*(f),f\bigr)\bigr]^{-1}.\tag{3.14}$$

By combining all the above with the identity $D\mathcal{K}(f)=D\mathcal{N}\bigl(\mathcal{U}^*(f)\bigr)D\mathcal{U}^*(f)$, we obtain an explicit expression for the derivative of the transformation $\mathcal{K}$.

4. Organization of the Proof

In this section, we describe how the goal of proving Lemma 1.4 and Lemma 1.6 is reduced to the goal of successfully executing a sequence of 16 computer programs. This description covers all major steps in our main programs (files `program_name.p` in `Section.6`), excluding definitions. The lower level procedures and functions will be explained in the next section.

We start by noting that the assertion of Lemma 1.4 is proved for a polynomial f_0 and a set $V = f_0 + V_0$ that are given explicitly: The coefficients of f_0 can be found in the file `approx.t`, and V_0 is specified in `nhood.v`. Other quantities are "constructed" in the course of the proof. In the case of a function f in $\mathcal{B}_\rho$, this means that we determine upper and lower bounds on the first $N+1$ Taylor coefficients $c_n(f)$, and an interval $[0,b]$ containing the norm $e_\rho(f)$ of the higher order terms. The resulting $N+2$ intervals define a convex neighborhood of f in the space $\mathcal{B}_\rho$. Any file mentioned here, whose name ends in "`.v`", specifies a convex set of this type.

The various constants like $\beta^2 = 2^{-5/3}$ that appear in the definitions of $\mathcal{M}$ and $\mathcal{K}$ are bounded from above and below, and these bounds are saved, by running the program `initial`. The same program also reads f_0 and writes the set $\{f_0\}$ as `approx.v`. Another quantity that is easy to construct is the matrix $P_{29} - P_{29}D\mathcal{N}(P_{100}f_0)P_{29}$ which enters the definition of $\mathcal{M}$. This matrix is constructed and inverted by running the program `matrix`, which saves the result as `minv.v` and `m.v`, respectively. As was shown in Corollary 3.2, the invertibility of M' implies that $\mathcal{M}$ is well defined and of class C^∞ on $\mathcal{B}_{10}$.

The parts of Lemma 1.4 that remain to be proved are

($P1$) The image of V under the map $\mathcal{M}$ is contained in V.

($P2$) $\|D\mathcal{M}(f)\|_{10} < 1$, for all $f \in V$.

($P3$) Every function f in $\mathcal{M}(V)$ is of type Z.

An important preparatory step in the proof of ($P1$) and ($P2$) is to construct the image κ of the set $\{f \in \mathcal{B}_{10} : P_N f = 0, e_{10}(f) \le 1\}$, with $N = 1000$, under the convolution H_λ. This is done by running the program `bounds`, which simply calls the procedure `vconvinit` (described later) and saves the result as `kappa.v`. The convex set containing κ is used again by `mmrest`. This program, together with `mmfirst`, estimates the difference $h_0 = \mathcal{M}(f_0) - f_0$ and saves the result in `difference.v`. The proof of ($P1$) is completed by running `contract`, which first constructs the terms in $\mathcal{M}(f_0 + h)$ that are linear or bilinear in h, where h is an arbitrary function in V_0. Then, the result is added to the neighborhood of h_0 found earlier, in order to obtain a set containing $\mathcal{M}(V) - f_0$. This set gets saved as `contracted.v`, after it is verified that it is contained in V_0.

We note that our direct method for verifying property ($P1$) is not commonly used in computer–assisted proofs. Any straightforward implementation of a contraction will in general fail to yield such a result. What makes the method work here is that we construct the linear term in the (trivial) Taylor expansion about f_0 of $\mathcal{M}(f)$ in a way that already takes into account major cancellations, by using the last expression in (3.5).

For the same reason, the task of proving $(P2)$ is unusually simple. The operator norm of $D\mathcal{M}(f)$, for $f \in V$, can be estimated accurately in one step, by applying $D\mathcal{M}(f)$ to the unit ball $\|h\|_{10} \leq 1$ (which can be represented as a convex set in $\mathcal{B}_{10}$ of the type described at the beginning of this section) and bounding the norm of the result. This estimate is carried out by running the program `dmmnorm`, which also uses the representation (3.5) for $D\mathcal{M}(f)$. The resulting bound turns out to be less than 1/5.

One of the steps in our proof of $(P3)$ consists in checking that for $k = 0, \ldots, 79$, each function in $\mathcal{M}(V)$ has a zero in a given small interval I_k centered at $-z_k$, where z_k is the k^{th} non–integer value listed in `zeros.t`. The program `zcheck`, which performs this task, first adds f_0 to the previously determined neighborhood of $\mathcal{M}(V) - f_0$, in order to get a set $\tilde{V}$ containing $\mathcal{M}(V)$. Then it is verified that for each $f \in \tilde{V}$, the values of f changes sign on each of the intervals I_k. The set $\tilde{V}$ is saved as `fixpt.v`.

Using the intervals I_k mentioned above, for $k \geq 5$, the program `checkgap` verifies that if $-x_k$ and $-x_{k-1}$ are arbitrary points in I_k and I_{k-1}, respectively, then $\sqrt{x_k} - \sqrt{x_{k-1}} < 4/3$. The same program also verifies the trivial bound [11, equation (3.2)], which was used in the proof of (1.2).

The proof of $(P3)$, and thus of Lemma 1.4, is completed by running the program `divide`. Here, every function $f \in \tilde{V}$ is divided by a polynomial p_f of degree 5 that vanishes in each of the intervals $I_0, \ldots, I_4$ at some zero of f. By constructing the quotient f/p_f on an interval $[-x, 0]$ containing I_4, it is verified that f has exactly 5 zeros between $-x$ and 0.

The first step in the proof of Lemma 1.6 is the construction of an approximate fixed point f_2, which is of the form $\mathcal{U}(u,p)$ for some polynomial p, and whose zeros all lie on the negative real axis. The starting point is a polynomial f_1 whose zeros are numerical approximations to the first 5000 zeros of f^*. The program `mkpoly` reads these approximate zeros from `zeros.t`, constructs the Taylor coefficients of f_1, and writes the result to `poly.v`. Separate, more accurate bounds on the first 200 coefficients of f_1 are determined by running `mkcoeff`. These bounds are saved in `pcoeff.v`.

Despite its high degree, the polynomial f_1 is not a very good approximation to the fixed point f^*. (Truncating the canonical product for f^* at the k–th zero introduces an error of the order $k^{-2/3}$ in the Taylor coefficients.) In order to find a better approximation, a sequence of transformations similar to $\mathcal{K}$ is applied to f_1. Each of these transformations is of the form $f \mapsto \mathcal{N}\bigl(\mathcal{U}(u,f)\bigr)$, where u is some vector in $\mathbb{R}^4$ close to $u^*(f)$. The reason for not using the transformation $\mathcal{K}$ at this point is to avoid an unnecessary degradation of our bounds. The program `kkiter`, which implements these steps, starts by initializing the convolution procedure as in `bounds`, but with the new value $\rho = 6$. The result is saved in `kkappa.v`. Then the abovementioned iteration is carried out. Part of each step is to verify that none of the zeros crosses the origin. The result after 80 steps is (a convex neighborhood of) the function f_2 referred to in Lemma 1.6. It is saved in the file `kkin.v`. At the end, the "full" transformation $\mathcal{K}$ is applied to f_2, and the result is written to `kkout.v`. To be more precise, it is not verified until later, at the beginning of the program `dkkmake` (see the description of `ustar` in Subsection 5.7), that f_2 lies in the domain of $\mathcal{K}$ as described in Corollary 3.4.

The operator Γ mentioned in Lemma 1.6 maps f to γf whenever $P_{28}f = 0$, and its action on $P_{29}\mathcal{B}_6$ is given by the 30×30 matrix with elements $\Gamma_{ij} = 6^j j! G_{ij}$, where G is the matrix defined in `g.m`. However, instead of computing the numbers Γ_{ij}, we change the norm on $\mathcal{B}_6$ such that only the matrices G and G^{-1} are needed. The program `kkmore` constructs the inverse of G and the ball $W = \{f \in \mathcal{B}_6 : \|\Gamma^{-1}[f - f_2]\|_6 \le 5\delta\}$, where δ is determined as an upper bound on the norm $\|\Gamma^{-1}[\mathcal{K}(f_2) - f_2]\|_6$. The results are saved in `ginv.m` and `kkball.v`, respectively.

The parts of Lemma 1.6 that remain to be proved are

($P4$) $\|\Gamma^{-1} D\mathcal{K}(f)\Gamma\|_6 \le \frac{4}{5}$ for every function f in W.

($P5$) W contains the set $\mathcal{M}(V)$.

The proof of ($P4$) follows a standard procedure. First, we determine the 30×30 matrix D, whose first 29 columns represent (sets containing) the images under $D\mathcal{K}(f)$ of the monomials $z \mapsto z^n$, with n ranging from 0 to $N = 28$, and whose last column represents the image of the higher order "ball" characterized by the conditions $P_N h = 0$ and $e_6(h) \le 1$. Here, f stands for an arbitrary function in W. The matrix D is constructed by running the program `dkkmake`, which saves it in transposed form in `dkkt.m`. These bounds are then used in `dkkcheck` in order to verify ($P4$). The same program also checks the inclusion ($P5$), which is the last step in our proof of Lemma 1.6.

All steps described in this section have been carried out by successfully running the indicated programs on a SPARCstation–10 by Sun Microsystems, Inc.

5. Lower Level Operations

The programs mentioned in the last section constitute the top level in a hierarchy of programs, procedures, and functions, which reduce the proof of our Theorems to a large number of simple integer operations and comparisons. All the necessary instructions have been written in a dialect [17] of the programming language Pascal [16]. The present section is meant to be a guide to the "real" and most detailed description of our proof — the source code of our programs. We will explain the basic strategies, and, in general terms, the intended action of each building block (procedure or function). For a basic introduction to computer–assisted proofs we refer to [15]; see also [13,14] and other references given in [15].

Each of the following subsections describes a collection of procedures and functions that are intended to be included (by programs) as a whole. These collections will be presented in increasing order of abstraction: At any given level, a procedure will be defined only in terms of procedures of minimally lower or equal level; and the same applies to the types of data on which these procedures act. All global constants, types, and variables that are shared by the procedures in a given collection are listed in the corresponding include–file (whose name ends in "`.i`"; see below).

Let us start by giving a definition of what we call a bound in the context of computer–assisted proofs [15]. Denote by $\mathcal{P}(\Sigma)$ the set of all subsets of a set Σ. Let F and G be maps from $D_F \subset \mathcal{P}(\Sigma)$ and $D_G \subset \mathcal{P}(\Sigma)$ to $\mathcal{P}(\Sigma')$, where Σ and Σ' are given sets.

Definition 5.1. *G is called a bound on F if $D_F \supset D_G$, and if $F(B) \subset G(B)$ for all $B \in D_G$.*

In order to estimate e.g the norm function on $\mathcal{B}_\rho$, we will bound the associated set–map F which assigns to a set $B \in \mathcal{P}(\mathcal{B}_\rho)$ the set of all values $\|h\|_\rho$ as h ranges over B. For practical reasons, the domain and range of our bound G will always be chosen within a finite collection of "standard sets" that can be represented on the computer with a given data type. For example, the standard sets in $\mathbb{R}$ will be intervals whose endpoints belong to a finite set $\Re$ of real numbers defined below. The standard sets for a finite product of spaces Σ_i will (by choice) always be corresponding products of standard sets for the factors Σ_i.

5.1. Operations on the Level of Real Numbers

Our basic numerical data type (besides integers) is the type `hreal`, which is used to represent fixed–length floating point numbers in the base 10^4. Such a number can be written as

$$r = \sigma \sum_{k=0}^{m} d_k 10^{4(E-k)}, \tag{5.1}$$

where $\sigma \in \{-1, 0, 1\}$ is the sign, E is the exponent, and $d_0, d_1, \ldots, d_m$ are the digits of r with values in $\{0, 1, \ldots, 9999\}$. The length m of the mantissa is fixed to some constant value `hdim` at the beginning of each main program.

Given $m > 1$, the set $\Re$ of representable numbers (reps) is now defined as the set of all real numbers r that can be written in the form (5.1), with $d_0 \neq 0$ whenever $\sigma \neq 0$, and with $|E| \leq 10^4$.

A list, with a short description, of all procedures and functions that operate directly on reps is given in the file `reps.i`. This includes conversions from integers to reps (`irset`), conversions between reps and decimal strings (`srset`, `rst`) used for input and output, multiplication and division by integers (`irmult`, `irdiv`), the standard arithmetic operations (`rsum`, `rdiff`, `rprod`, `rquot`), the square root function (`rsqrt`), and several procedures that deal with upper and lower bounds (see below).

Every one of these procedures (except `rst`) either returns a result that lies explicitly in $\Re$, or else it sets the appropriate flag `underflow` or `overflow`, which is a global Boolean variable. In order to ensure that only results with $\sigma = 0$ or $d_0 > 0$ are returned, some of the procedures call `rnormalize`. We note that global quantities (types, constants, variables) that are shared by all programs, are defined or declared in the file `types.h`. Examples are the limit `exmax` $= 10^4$ used in the definition of $\Re$, and the Boolean constants `up` $=$ `true` and `down` $=$ `false`.

An important property of the abovementioned procedures is that if their input is restricted to values in $\Re$, then all digits of the returned result coincide with the corresponding digits of the exact result. Thus, the action performed is equivalent to first computing the result exactly, and then truncating it to $m+1$ digits. In case the truncated digits were not all zero, the procedures set a flag named `trunc`. Here, we assumed that `exmax` is smaller than the largest (as well as the negative of the smallest) available `integer`, and larger than the value of `hdim`. These conditions are met in all our programs.

Each of the procedures mentioned above is usually called from inside a similarly named procedure (`ir_mult`, `ir_div`, `r_sum`, `r_diff`, `r_prod`, `r_quot`, `r_sqrt`) which then uses the returned value of `trunc` in order to find an upper or lower bound on the exact result. The type of bound is specified in the first argument, which can be set to either `up` or `down`. For example, if $r > s$ are positive reps, then `r_diff(down, r, s, t)` will return in t the lower bound on $r - s$ obtained from `rdiff(r, s, t)`; and `r_diff(up, r, s, t)` will return either the same value, if `trunc` has not been set, or else the next larger number in $\Re$. In the last case, the procedure `rtowinfty` is used, which determines for a given number in $\Re$ its closest neighbor in $\Re$ in the direction away from zero; if this is not possible, the `overflow` flag is set. An analogous procedure `rtowzero` determines the closest nonzero neighbor in $\Re$ in the direction towards the origin, if possible, or else it sets the `underflow` flag.

Underflow will be handled later, by the procedures discussed in the next subsection. Two reps that are used in this context are the smallest positive number in $\Re$ and its negative, which are returned by the functions `minhreal` and `negminhreal`, respectively. Two other procedures that use these number are `rup` and `rdown`, which determine the successor and predecessor, respectively, of a number in $\Re$; if this is not possible, the `overflow` flag is set.

Comparison between two reps is done by using an integer `irdiff(r, s)`, which has the same sign (0 or ± 1) as the difference $r - s$.

5.2. Elementary Bounds Involving Scalars

As mentioned after the Definition 5.1, we choose as standard sets in $\mathbb{R}$ non–empty closed intervals $[a, b]$ whose endpoints a and b are numbers in $\Re$. Such sets will also be referred to as scalars. Scalars that are often used are `sone` $= [1, 1]$, `stwo` $= [2, 2]$, and `shalf` $= [0.5, 0.5]$. They are defined as global variables in the procedure `sinit`, which has to be called before any other procedure discussed here.

Setting a scalar s to $[0, 0]$ is done by calling `szero(s)`. A conversion procedure, called `ssset`, converts a decimal string to the shortest scalar (interval) containing the floating point number represented by the string; if there is no such scalar, then the `overflow` flag is set. This procedure is used e.g. by `sread` in order to read a scalar from a file. Writing a scalar to a file is done by calling `swrite`. Two other trivial but useful interval operations are `sunion` and `sinter`, which return the convex hull and the intersection, respectively, of their first two scalar arguments. An empty intersection in `sinter` results in an error message.

For comparisons between two scalars we use the Boolean functions `ssub`, `sgt`, `sge`, `slt`, and `sle`. Here, `ssub(r, s)` is `true` iff r is a subinterval of s. The other functions represent the relations "$>$", "$\geq$", "$<$", and "$\leq$". For example, `sgt(r, s)` is `true` if and only if every number in r is larger than every number in s. The remaining three relations are defined similarly. As for the relation "$=$", only comparisons with $[0, 0]$ will be needed; the corresponding function is `seq0`.

Bounds on the multiplication and division by integers, and on the standard arithmetic operations, are provided by the procedures `ismult`, `isdiv`, `ssum`, `sdiff`, `sprod`, and `sqot`. Here, bounds are to be understood in the sense of Definition 5.1. For example, `squot(r, s, t)` returns in t a scalar that contains all values that can be obtained by dividing any number x in r by any number y in s; or if a domain violation occurs (e.g. if s contains zero), then an error message is generated. Domain violations include numeric overflow, but not underflow. When a reps–procedure causes an underflow, it is still possible to give upper and lower bounds on the correct result by using the values `minhreal`, `negminhreal`, and 0. The procedures and functions discussed here takes advantage of this fact, if necessary, and then reset `underflow` to `false`. This is done by calling `snormalize`.

We note that our choice of standard sets is such that two bounds can be composed the same way as the original maps, provided that the range of the first bound is contained in the domain of the second. This makes it possible to use the simple bounds discussed so far in order to implement bounds on more complex maps. The domains of these more complex bounds are defined in terms of the domains of the various procedures and functions they use. In practice, this means that if a bound G is applied to input B from the appropriate standard set, and it returns without having generated (either by itself, or by calling other procedures) any error messages or numeric overflow, then B lies in D_G by definition, and $G(B)$ lies in the correct standard set. One type of error message is the numeric overflow. It can only occur in the reps procedures and functions described in the last subsection; but once it occurs, the `overflow` flag stays set. The corresponding error message is printed by the procedure `sdone`, which is called at the end of each program.

The basic set of scalar operations, as listed in the file `scalar.i`, also includes the functions unary minus, absolute value, square root, factorials (of non–negative integers), and integer powers. The procedures which provide bounds on these functions are `sneg`, `sabs`, `ssqrt`, `isfactorial`, and `ispower`. The last one in this list is an example of a composed bound. In particular, `ispower(r, 4, t)` finds a scalar t that contains the fourth power of every number in a given scalar r, by calling first `sprod(r, r, s)` and then `sprod(s, s, t)`.

Finally, for the sake of completeness: `sabs0(r, s)` returns a scalar s that contains 0 and $|x|$ for every x in a given scalar r, `senlarge(s)` enlarges s minimally on both sides, `shrink(s)` shrinks s to a subinterval of length zero, and `ssymm(r, s)` is used to convert a non–negative scalar $r = [a, b]$, to a symmetric scalar $s = [-b, b]$.

5.3. Additional Scalar Functions

The bounds discussed here are used only by the programs `initial` and `checkgap`, which include them through the file `sfun.i`. In addition, they are only called a few times. Thus, in implementing these bounds, we have traded efficiency (with respect to both accuracy and speed) for simplicity. In particular, the task is always reduced (by means of trivial identities) to a point where a Taylor expansion can be used, with a remainder that is smaller in absolute value than a fixed constant times the last explicitly bounded term. Bounds on the individual terms in the expansion, and on their sum, are obtained by using the procedures mentioned in the last subsection.

The exponential function is bounded by the map defined through the procedure `sexp`. Here, the identity $\exp(nx) = [\exp(x)]^n$ is used, in order to restrict the Taylor expansion for $\exp(x)$ to cases where $|x| \le$ `hdim`$/136$.

Since a non–integer power of 2 enters the definition of $\mathcal{N}$ and related transformations, we also determine a scalar which contains $\ln(1/2)$. This scalar is constructed by the procedure `slnhalf`.

The procedure `sarctan` provides a bound on the function arctan. In this case we apply the identity $\arctan(x) = 2^n \arctan(x_n)$, where x_n is the image of x under the n^{th} iterate of the map $x \mapsto x/\bigl(\sqrt{1+x^2}+1\bigr)$, with n sufficiently large such that $|x_n| < 1/10$. Then a Taylor expansion is used as indicated above. Given the bound on the function arctan, one easily obtains a bound on the function arcsin. This is done by the procedure `sarcsin`.

5.4. Bounds Involving Vectors

In this subsection we consider bounds on maps to and from the Banach spaces $\mathcal{B}_\rho$. Here, ρ is any positive real number — not necessarily an element of $\Re$. In order to define the standard sets for $\mathcal{B}_\rho$, which will henceforth be referred to as vectors, we need to fix a positive integer N. Then a vector is defined in terms of $N+2$ scalars $v_0, v_1, \ldots, v_{N+1}$, with v_{N+1} of the form $[0, b]$, and is given by the set

$$v = \bigl\{ f \in \mathcal{B}_\rho : \; c_k(f) \in v_k \text{ for } 0 \le k \le N, \; e_\rho(f) \in v_{N+1} \bigr\} . \tag{5.2}$$

Here, $c_k(f)$ denotes the k^{th} Taylor coefficient of f at zero, and $e_\rho(f)$ is the bound on the higher order terms of f defined in (2.2). In our programs, the corresponding data type `vector` is an array of scalars indexed by $\{0, 1, \ldots,$ `vdim`$\}$, where `vdim`$= N+1$ is a constant defined at the beginning of each program.

One of the easiest bound to implement is that on the sum of two vectors. The procedure `vsum` determines a vector w containing the sum of two vectors $u+v$ by calling `ssum(u[k],v[k],w[k])` for $k = 0, \ldots,$ `vdim`. This clearly defines a bound, in the sense of Definition 5.1, on the set–map associated with the function "+" from $\mathcal{B}_\rho \times \mathcal{B}_\rho$

to $\mathcal{B}_\rho$. The domain of this bound is defined in terms of the domain of the bound represented by `ssum`, as mentioned earlier. A related procedure `vadd` differs from `vsum` the same way as the assignment $u+v \to v$ differs from $u+v \to w$.

Two other procedures that work independently of the choice of the space $\mathcal{B}_\rho$ are `vdiff` and `vsprod`, which provide bounds on the difference of two vectors, and on the product of a vector with a scalar. Further examples in this category are the procedure `vzero` which returns the vector $\{0\}$, the Boolean function `vsub` which determines whether its first argument is a subset of the second, the procedure `vpositive` which bounds the trivial projection from $\mathcal{B}_\rho$ to $\mathcal{B}_\rho^0$, and the vector–analogue `venlarge` of `senlarge`.

Most procedures that operate on vectors in $\mathcal{B}_\rho$ require as part of their input a scalar that contains the value of ρ. Consider e.g. the procedure `vnorm` which bounds the function $f \mapsto \|f\|_\rho$. If r is a scalar containing ρ, and if v is a vector in $\mathcal{B}_\rho$, then `vnorm(v,r,s)` returns in s a scalar that contains $\|f\|_\rho$ for every function f in v.

We note that the inclusion map from $\mathcal{B}_\rho$ to $\mathcal{B}_a$, where $a \le \rho$, admits an optimal bound that is as trivial as the map itself: the bound represented by the identity procedure "no operation". This fact, which is due to our choice of normalization factors in the definition of e_ρ, is frequently used by our programs. For example, a `vector` can be passed from one program to the next (using `vwrite` and `vread`) and thereby undergo an implicit conversion from a vector $u \subset \mathcal{B}_{10}$ to a vector $v \subset \mathcal{B}_6$ which contains the original set u.

A consequence of the above is the fact that the scaling operator $f \mapsto f(\kappa.)$ from $\mathcal{B}_\sigma$ to $\mathcal{B}_\rho$ can be bounded sharply without any knowledge about σ or ρ, other than the necessary condition $|\kappa\rho| \le \sigma$. This bound is implemented by the procedure `vscale` and its helper `vpowers`.

Two frequently used subsets of $\mathcal{B}_\rho$, besides $\{0\}$, are the ball $\|f\|_\rho \le b$ and the set $\{f_\alpha\}$, where $f_\alpha(x) = \exp(\alpha x)$. Here, b and α are given real numbers satisfying $b > 0$ and $|\alpha\rho| < 1$. Vectors containing these two sets are returned by `vball` and `vexp`, respectively. Actually, `vexp` does little more than calling the procedure `pexp` which bounds the Taylor coefficients of $P_d f_\alpha$ for any given $d \le N+1$. Our reason for introducing this additional procedure is that a bound on $P_3 f_\alpha$ is also needed elsewhere. The collection `vector.i` contains other auxiliary procedures of this type (`pconv`, `pder`, `pprod`, `pscale`), but we shall not always mention them explicitly.

The procedures discussed next use the estimates given in Section 2.

The translation operator $(f,x) \mapsto f(.+x)$ and the evaluation functional $(f,x) \mapsto f(x)$ are bounded by the maps defined in `vtrans` and `sval`. These bounds simply implement the estimates from Subsection 2.1. We note that the procedure `vtrans` can be instructed to bound only the first $d+1$ Taylor coefficients of the translated function. This is done by specifying a value $d <$ `vdim` in the first argument. An analogous feature has been added to several other procedures, in order to save computation time in cases where the map is followed directly by a projection P_d. This occurs e.g. in the definition of $\mathcal{K}$, with $d=3$.

A bound on the map $(f,g) \mapsto fg$ from $\mathcal{B}_\sigma \times \mathcal{B}_\tau$ to the appropriate space $\mathcal{B}_\rho$ is given by the procedure vprod. Its arguments are standard sets containing σ, τ, f, g, and fg (the output). The estimates used here are given in Proposition 2.4. A similar but more efficient procedure vsqr applies if $f = g$. Two short procedures vder and vdquot implement the estimates of Proposition 2.6 on the derivative and on a difference quotient.

The convolution operator $H_\lambda:\ f \mapsto J^{-1}\bigl(g * (Jf)\bigr)$, where $\bigl(Jf\bigr)(x) = f(x^2)$, and where g is the normalized Gaussian with covariance $c = 2\lambda > 0$, is bounded by using the estimates from Subsection 2.4. The crucial part here is to estimate the effect of the higher order term $e_\sigma(f)$ of $f \in \mathcal{B}_\sigma$ on the Taylor coefficients of the image. Fortunately, since H_λ is linear, it suffices to perform this estimate only once (for a given σ) and for a general function f satisfying $P_N f = 0$ and $e_\sigma(f) = 1$. The restriction of H_λ to this "higher order ball" is bounded by the procedure vconvinit. This procedure uses both Proposition 2.8 and Proposition 2.9, and takes the better of the two estimates for each Taylor coefficient of $H_\lambda f$. The resulting vector (often referred to as κ) is one of the required arguments in the procedure vconv which implements the bound on the entire map H_λ .

5.5. Other Linear Spaces

The operations discussed here are included through the file linear.i. Most of them involve only finite–dimensional vector spaces. The exceptions are described at the beginning and end of this subsection.

In the few cases where we need to be able to represent vectors with two different values of N within the same program, we use a data type ltuple besides the type vector. A ltuple is an array of scalars indexed by $\{0, 1, \ldots, \texttt{lmax}\}$, and lmax is always chosen to be smaller than the maximal subscript vdim of a vector. A conversion procedure called lvconvert, assisted by lvho (which bounds the higher order term), determines a "regular" vector that contains the standard set associated with a given ltuple. Conversion in the opposite direction is done with the procedure vlconvert, assisted by vlho.

The regular usage of the type ltuple is for the representation of standard sets in $\mathbb{R}^L$, where $L = \texttt{lmax} + 1$. A standard set in $\mathbb{R}^L$, also referred to as L–tuple, is defined to be a direct product of L scalars.

Given a point $\omega = (\omega_0, \omega_1, \ldots, \omega_{L-1})$ in $\mathbb{R}^L$, whose coordinates ω_i are all positive, we define a norm on $\mathbb{R}^L$ by setting $\|u\|_\omega = \max\{\omega_i^{-1}|u_i|:\ 0 \le i < L\}$ for all $u \in \mathbb{R}^L$. A bound on this norm is given by the procedure lnorm, whose first argument is expected to be a L–tuple w containing ω. If r is a scalar containing a given real number $b > 0$, then lball(r,w,u) will return in u a L–tuple that contains the ball of radius b for the norm defined above.

Bounds on the elementary vector space operations "sum", "difference", and "product with scalars", are implemented by the procedures lsum, ldiff, and lsprod, respectively.

Consider now the space $\mathbb{R}^{L\times L}$, representing linear operators on $\mathbb{R}^L$. Following our convention for product spaces, we define the corresponding standard sets to be $L \times L$ matrices whose entries are scalars.

Our procedures for reading and writing such standard sets (type `matrix`) are `mread` and `mwrite`. Bounds on the inverse and on the usual products are given by `minverse`, `lmprod` (matrix times L–tuple) and `mprod` (matrix times matrix). A matrix–norm is defined by regarding matrices as linear operators on the Banach space $\mathbb{R}^L$, equipped with the norm $\|.\|_\omega$ defined above. This operator norm is bounded in `mnorm`.

If a given $L \times L$ matrix is extended to an infinite "square matrix" by filling in 1's along the main diagonal and 0's everywhere else, then the result defines a linear operator M on $\mathcal{B}_\rho$. A bound on the map $f \mapsto Mf$ is given by the procedure `vmprod`.

5.6. Bounds on the Terms of $D\mathcal{M}$

The procedures mentioned here are called by the programs `contract` and `dmmnorm`. A list of global variables used by these procedures is given in the file `hiemaps.i`. Among these global variables are scalars that are expected to contain the values $\beta^2 = 2^{-5/3}$, $c = 2\lambda$, $\rho = 10$, σ, and τ; see equation (3.2). Two other global variables that are expected to be defined properly are the `vector vkappa` (determined by the procedure `vconvinit`) and the `matrix m` (determined by the program `matrix`).

The procedure `nn` provides a bound on the transformation $\mathcal{N}$ defined in equation (1.1), by appropriately composing the procedures `vscale`, `vsqr`, and `vconv`. The procedure `dnn` which is used to bound the derivative of $\mathcal{N}$ is very similar, since $\mathcal{N}$ is quadratic. We note that the parameter ρ enters `vconv` only implicitly through the bound `vkappa`.

Since the map $\mathcal{M}$ is quadratic as well, it suffices to implement a bound on its derivative. This is done by the procedures `dmm1`, `dmm2`, and `dmm3`. The three parts correspond to the three terms in the final expression for $D\mathcal{M}$ in equation (3.5). In particular, the procedure `dmm2` bounds the linear operator M (`matrix m`) in its last step.

5.7. Bounds on the Terms of $\mathcal{K}$ and $D\mathcal{K}$

The procedures discussed here are used by the programs `kkiter` and `dkkmake`. In both of these programs, the maximal subscript `lmax` for the type `ltuple` is 3. Among the global variables are the scalars mentioned above (but now with $\rho = 6$) and the bound `vkappa`. The latter has to be reconstructed for the new value $\rho = 6$. A complete list of global variables is given in the file `kkmap.i`.

Several of the procedures considered in this subsection will skip certain instructions whenever a global Boolean variable called `normalize` has the value `false`. This mode corresponds to the approximate version of $\mathcal{K}$ that was mentioned in the description of the program `kkiter`. We shall only discuss the full version here.

Two other global variables, the L–tuples `lnormal` and `umax`, are initialized in the procedure `kkinit`. The scalar components of `lnormal` contain (as is verified later) the first four Taylor coefficients of the fixed point f^* for $\mathcal{M}$. The scalars `umax[1]` and `umax[2]` contain the values v and $1+v$ which define the set $U \subset \mathbb{R}^4$ in (3.6). These scalars are used in the Boolean function `uok`, which determines whether a given L–tuple (the first argument) lies in the set U_f for every f in a given vector (the second argument) of $\mathcal{B}_\rho$.

The procedure `ustar` provides a bound on the map $f \mapsto u^*(f)$ and verifies the assumptions of Corollary 3.4. The neighborhood W' mentioned in this Corollary is the interior of the set obtained by applying `venlarge` to the input vector `v` corresponding to f, and the set B is given by the L–tuple returned by `ustar`. The set B is obtained from the procedure `findu`, which applies the contraction mapping principle to the Newton map associated with F, where $F(u) = \mathcal{F}(u, f)$; see (3.9) for the definition of $\mathcal{F}$. This step also involves a bound on the tangent map $u \mapsto \bigl(F(u), DF(u)\bigr)$, provided by the procedure `df`, which uses the formula (3.11) for the derivative $D_1\mathcal{U}$.

A bound on the map $\mathcal{K}$ on $\mathcal{B}_\rho$ is defined by the procedure `kk`. The three last steps in `kk` are the canonical bound on $\mathcal{N}$, regarded as a map from $\mathcal{B}_r$ to $\mathcal{B}_\rho$, where $r = \beta^2\tau$. They are preceded by a bound on the "unpack operator" $\mathcal{U}^*$, which is the composition of $f \mapsto (u^*(f), f)$ with $\mathcal{U}$. The corresponding procedure `unpack` first calls `findu`, then determines a bound (in `ve`) on the exponential factor that appears in the definition (1.8) of $\mathcal{U}$, and then passes everything to the procedure `dunpack1` which does the rest. We note that `dunpack1` bounds the product in (1.8) as indicated in the proof of Proposition 3.3, but with $s = 1/(4b_1)$.

The derivative $D\mathcal{K}(f)$, where f is an arbitrary function in a given standard set v of $\mathcal{B}_\rho$, is bounded by using the procedure `dkk`. In order to increase efficiency, several quantities that depend only on v are determined beforehand, and passed to `dkk` as additional arguments. These quantities (together with the names used in the declaration of `dkk`) are the L–tuple `u` returned by `ustar`, the vector `ve` mentioned above, a `vtuple` `vv` representing a bound on the operator $\mathcal{V}(f)$ defined in (3.14), and a vector `vus` (obtained easily from `vv`) containing the result of unpacking v to w and then scaling w with β^2.

The type `vtuple` is an array of vectors, with indices ranging from 0 to `lmax` = 3. It is used to represent standard sets for the space of linear operators form $P_3\mathcal{B}_r$ to $\mathcal{B}_r$, which can be identified canonically with $(\mathcal{B}_r)^4$.

Both `ve` and `vv` are determined by the procedure `dkkinit`. The partial derivatives (3.11) of $\mathcal{U}$, which enter the definition of $\mathcal{V}$, are bounded as indicated in the proof of Proposition 3.3. The last step in the construction of `vv` is to call the procedure `vvmtprod`, which bounds the product in equation (3.14).

The action performed by `dkk` consists in composing a bound on $D\mathcal{U}^*(v)$ with a bound on $D\mathcal{N}(w)$, where $w = \mathcal{U}^*(v)$. The first bound uses the procedures `dunpack1` and `dunpack2`, which correspond to the factors $D_2\mathcal{U}(u^*(f), f)$ and $(1-P_3)[1 - \mathcal{V}(f)P_3]$ in equation (3.13). The second part is equivalent to `dnn`, with a call to `vscale` omitted since its result is already available in the abovementioned vector `vus`.

This concludes our description of how the steps outlined in Section 4 are reduced to a sequence of smaller steps that can be carried out by a computer running our programs. For details, we refer to the source code of these programs.

References

[1] G.A. Baker, *Ising Model with a Scaling Interaction.* Phys. Rev. **B5**, 2622–2633 (1972).

[2] F.J. Dyson, *Existence of a phase transition in a one–dimensional Ising ferromagnet.* Comm. Math. Phys. **12**, 91–107 (1969).

[3] P.M. Bleher, Ja.G. Sinai, *Investigation of the critical point in models of the type of Dyson's hierarchical model.* Comm. Math. Phys., **33**, 23–42 (1973).

[4] P.M. Bleher, Ja.G. Sinai, *Critical indices for Dyson's asymptotically hierarchical models.* Comm. Math. Phys., **45**, 247–278 (1975).

[5] P. Collet, J.-P. Eckmann, *A renormalization group analysis of the hierarchical model in statistical physics.* Lecture Notes in Physics, **74**, Springer–Verlag, Berlin, Heidelberg, New York (1978).

[6] G. Gallavotti, *Some Aspects of the Renormalization Problems in Statistical Mechanics.* Memorie dell' Accademia dei Lincei **15**, 23–59 (1978).

[7] G. Benfatto, M. Cassandro, G. Gallavotti, F. Nicolò, E. Olivieri, E. Presutti, E. Scacciatelli, *Some probabilistic techniques in field theory.* Comm. Math. Phys., **71**, 95–130 (1980).

[8] K. Gawędzki, A. Kupiainen, *Non–Gaussian Fixed Points of the Block Spin Transformation. Hierarchical Model Approximation.* Comm. Math. Phys., **89**, 191–220 (1983).

[9] H. Koch, P. Wittwer, *A Non–Gaussian Renormalization Group Fixed Point for Hierarchical Scalar Lattice Field Theories.* Commun. Math. Phys. **106**, 495–532 (1986).

[10] H. Koch, P. Wittwer, *On the Renormalization Group Transformation for Scalar Hierarchical Models.* Commun. Math. Phys. **138**, 537–568 (1991).

[11] H. Koch, P. Wittwer, *A Nontrivial Renormalization Group Fixed Point for the Dyson–Baker Hierarchical Model.* Commun. Math. Phys., **164**, 627–647 (1994).

[12] A. Pordt, *Renormalization theory for hierarchical models.* Helv. Phys. Acta, **66**, 105–154 (1993).

[13] O.E. Lanford, *Computer–Assisted Proofs in Analysis.* Physica, **124 A**, 465–470 (1984).

[14] J.-P. Eckmann, H. Koch, P. Wittwer, *A Computer–Assisted Proof of Universality for Area–Preserving Maps.* Memoirs of the American Mathematical Society, **47**, 1–121 (1984).

[15] H. Koch, A. Schenkel, P. Wittwer, *Computer Assisted Proofs in Analysis and Programming in Logic: A Case Study.* Preprint mp_arc 94–394, to appear in SIAM review.

[16] D. Cooper, *Standard Pascal User Reference Manual.* W.W. Norton (1980).

[17] *Sun Pascal Reference Manual.* Sun Microsystems, Inc. (1990).