UCL DEPARTMENT OF
COMPUTER SCIENCE

Research Note
RN/13/20

Geosphere: Consistently Turning MIMO Capacity into Throughput

October 14, 2013

Konstantinos Nikitopoulos

Juan Zhou

Ben Congdon

Kyle Jamieson

Abstract

This paper presents the design and implementation of Geoshpere, a physical- and link-layer design for multi-cell access point-based MIMO networks that consistently improves network throughput. To send multiple streams of data in a MIMO system, prior designs rely on a technique called zero-forcing, a way of “nulling" the interference between the different spatial streams by inverting the effect of the wireless channel matrix. In many cases, when this channel matrix is well-conditioned, zero-forcing is highly effective, eliminating inter-stream interference. But measurements from our indoor wireless testbed network indicate that many of its links suffer from poorly-conditioned MIMO channel matrices. In these situations, zero-forcing techniques leave performance on the table, so Geosphere uses sphere decoding that can make fewer errors, and therefore can realize more of the MIMO capacity. To overcome the sphere decoder's computational complexity when signaling with dense constellations at a high rate, Geoshpere uses novel tree-search techniques that incorporate geometric reasoning about the constellation to reduce computational complexity by up to an order of magnitude. Thus Geosphere makes such an approach practical for the first time in a 4x4, 256-QAM MIMO system. Results from our WARP testbed show that Geosphere achieves average throughput gains of 2x in 4x4 MIMO systems and 47% in 2x2 MIMO systems, while simultaneously requiring up to nearly an order of magnitude less computation relative to the sphere decoder, bringing its computational demands in line with current systems already realized in ASIC.

This material is based on work supported by the European Research Council under Grant No. 279976

Geosphere: Consistently Turning MIMO Capacity into Throughput

Konstantinos Nikitopoulos, Juan Zhou, Ben Congdon, Kyle Jamieson
University College London

Abstract

This paper presents the design and implementation of Geosphere, a physical- and link-layer design for multi-cell access point-based MIMO networks that consistently improves network throughput. To send multiple streams of data in a MIMO system, prior designs rely on a technique called zero-forcing, a way of "nulling" the interference between the different spatial streams by inverting the effect of the wireless channel matrix. In many cases, when this channel matrix is well-conditioned, zero-forcing is highly effective, eliminating inter-stream interference. But measurements from our indoor wireless testbed network indicate that many of its links suffer from poorly-conditioned MIMO channel matrices. In these situations, zero-forcing techniques leave performance on the table, so Geosphere uses sphere decoding that can make fewer errors, and therefore can realize more of the MIMO capacity. To overcome the sphere decoder's computational complexity when signaling with dense constellations at a high rate, Geosphere uses novel tree-search techniques that incorporate geometric reasoning about the constellation to reduce computational complexity by up to an order of magnitude. Thus Geosphere makes such an approach practical for the first time in a 4×4, 256-QAM MIMO system. Results from our WARP testbed show that Geosphere achieves average throughput gains of $2\times$ in 4×4 MIMO systems and 47% in 2×2 MIMO systems, while simultaneously requiring up to nearly an order of magnitude less computation relative to the sphere decoder, bringing its computational demands in line with current systems already realized in ASIC.

1 Introduction

One of the most important challenges in modern wireless networks is to meet users' ever-increasing demand for throughput, and one way of meeting this demand is through a technique called *spatial multiplexing*. Networks

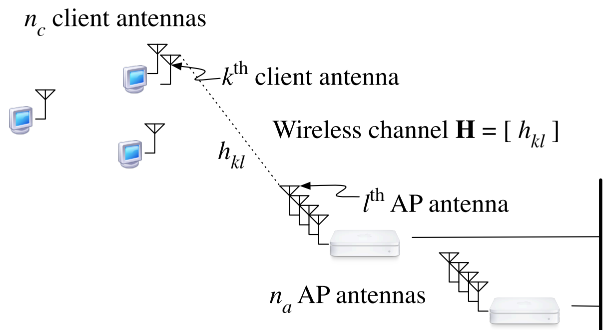

Figure 1: A MIMO wireless LAN with n_c client antennas and n_a AP antennas. Different clients and APs may transmit simultaneously, forming a MIMO system described by the channel matrix $\mathbf{H}$, whose entries characterize the wireless channel between a client's antenna and an AP's antenna.

that leverage spatial multiplexing increase capacity [30] and throughput by sending multiple streams of data from different transmit antennas. If enough receiving antennas hear the resulting mixture of signals and channel conditions are favorable, such systems can deliver these multiple data streams simultaneously, in the same frequency bands and geographical spaces. Since multiple transmit and receive antennas are required, these systems are called *multiple-input, multiple-output* or MIMO systems, and are ubiquitous in the design of wireless networks today.

The emergence of new applications such as video IP telephony (*e.g.*, Skype), wireless data backup, video surveillance, and direct video uploading (*e.g.*, Google Glass) is today shifting the ratio between downlink and uplink traffic in wireless networks towards the uplink. In this setting, mobile transmitters may simply send their own information streams to the access points (APs), which are connected by a wired network backhaul, as shown in Figure 1. While this frees clients from the need to cooperate with each other before sending, each of the receiving access point antennas then hears a tangled mixture $\mathbf{y}$ of the information sent from all transmit antennas $\mathbf{x}$ after it travels through the wireless channel $\mathbf{H}$, plus background noise $\mathbf{w}$:

$$\mathbf{y} = \mathbf{Hx} + \mathbf{w}. \tag{1}$$

The resulting *capacity* (*i.e.*, maximum theoretical throughput achievable) is then

$$C = \mathrm{E}\left[\log\det\left(\mathbf{I}_{n_r} + \frac{\mathrm{SNR}}{n_t}\mathbf{HH}^*\right)\right] \text{ bits/s/Hz},$$

where $\mathbf{H}$ is a matrix whose entries describe the wireless channel between each pair of client and AP antennas. In this work we consider the problem of improving uplink performance in a network served by a multi-antenna AP. This paper poses the question of how best to turn the above theoretical capacity gain into a practical throughput gain.

An oft-applied solution is a demodulation scheme known as *zero-forcing*. In order to decouple interfering streams a zero-forcing receiver left-multiplies the received vector $\mathbf{y}$ with the inverse of matrix $\mathbf{H}$, denoted $\mathbf{H}^{-1}$:

$$\mathbf{H}^{-1}\mathbf{y} = \mathbf{H}^{-1}\mathbf{Hx} + \mathbf{H}^{-1}\mathbf{w} = \mathbf{x} + \mathbf{H}^{-1}\mathbf{w} \tag{2}$$

Zero-forcing has been proposed as part of a way to accomplish spatial division multiplexing in SAM [28] and BigStation [32], null out aligned interference in IAC [9], and enable concurrent 802.11n transmissions in 802.11n$^+$ [19]. It works well in these systems, achieving multiplicative increases in capacity in line with expectations. But does it work consistently, on every link in a testbed deployment?

A closer look at Equation 2 indeed reveals room for improvement if we consider the *condition number* of $\mathbf{H}$

$$\kappa(\mathbf{H}) = \|\mathbf{H}\| \left\|\mathbf{H}^{-1}\right\|, \tag{3}$$

a metric from numeric linear algebra that measures the sensitivity of the linear system described by $\mathbf{H}$ to noise [18, Chp. 20]. $\kappa(\mathbf{H})$ is always greater than one. When the matrix $\mathbf{H}$ is well-conditioned ($\kappa^2(\mathbf{H}) <$ 20 dB[1] [1]), the noise term in Equation 2 above will be small:

$$\left\|\mathbf{H}^{-1}\mathbf{w}\right\|^2 \leq \left\|\mathbf{H}^{-1}\right\|^2 \|\mathbf{w}\|^2 = \frac{\kappa^2(\mathbf{H})}{\|\mathbf{H}\|^2}\|\mathbf{w}\|^2. \tag{4}$$

but when reflectors are located solely in the vicinity of one of the endpoints as shown in Figure 2, $\kappa(\mathbf{H})$ becomes large [31], $\mathbf{H}$ becomes almost singular, and its determinant becomes small in magnitude. Under these conditions, when the zero-forcing receiver left-multiplies the received signal by $\mathbf{H}^{-1}$ (Equation 2), it amplifies the background noise and interference $\mathbf{w}$, leading to bit-errors and decreased throughput [7].

[1] We use decibels relative to one to capture the wide range in variation of κ: κ^2 (dB) $= 20\log_{10}\kappa$ (linear scale).

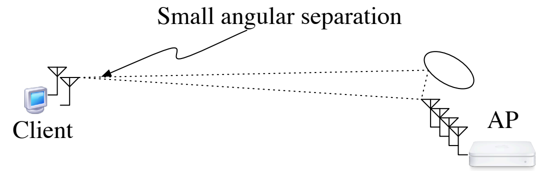

Figure 2: When reflectors are located solely in the vicinity of one of the endpoints (AP case shown here) of a MIMO link, the result is a very small angular separation of the energy arriving at the other end, and a poorly-conditioned channel matrix $\mathbf{H}$.

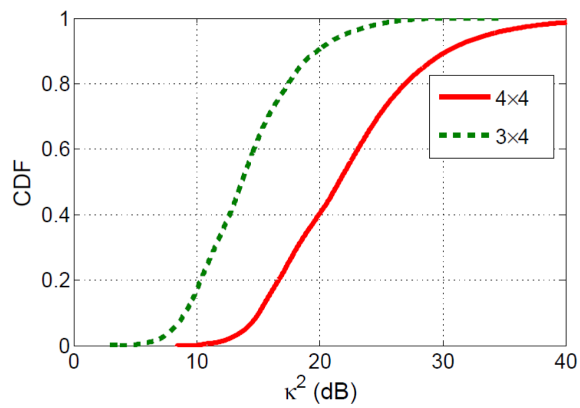

Figure 3: Cumulative distribution of the condition number $\kappa(\mathbf{H})$ (defined in Equation 3) in our wireless testbed.

But how often is the MIMO channel poorly-conditioned? Figure 3 shows measurements from our experimental wireless testbed (Figure 9 on page 7). If we use a generous 20 dB figure as a cutoff, these distributions show that a 3×4 MIMO channel is poorly-conditioned 10% of the time, while a 4×4 channel is poorly-conditioned 60% of the time, with all 4x4 channels having condition numbers of at least 8 dB. This is consistent with previous measurements of 2×2 MIMO channels in the 5 GHz band [17], and indicative of an opportunity to harness increased throughput.

This paper presents the design, implementation, and testbed evaluation of *Geosphere*, a system that closes the gap to capacity that zero-forcing's noise amplification opens. Geosphere uses a decoder that can achieve the theoretical maximum-likelihood (*i.e.*, the one that minimizes the probability of bit error), the *sphere decoder*.

The next section contains a primer on the sphere decoder, but in brief, the sphere decoder dramatically reduces the exponential (in terms of message length) asymptotic complexity of the maximum likelihood decoder by means of a tree search. On average, the sphere decoder achieves a computational complexity that allows system throughput on par with current wireless LAN speeds. For a point of reference, a 2005 hardware ASIC sphere decoder implementation [6] for a 4×4 16-QAM MIMO system in a Rayleigh fading channel achieves line rate over a 10 MHz frequency bandwidth. With advances in ASIC technology and the parallelizability of the sphere decoder by OFDM subcarrier, today's implementations can easily achieve line rates over 40 and 80 MHz wireless channels.

While the sphere decoder is practical today at slower 802.11g rates, the search for higher throughputs is driving the use of denser signal constellations. For example, 802.11ac devices already use 128- and 256-QAM constellations. Denser constellations are also prerequisite for adopting the promising new family of rateless codes [22, 10], which we discuss in further detail below (§6). However, the branching factor of the sphere decoder's tree search equals the size of the constellation. This means that denser constellations necessitate a larger search space for the sphere decoder, and a concurrent increase in computation, overwhelming current state-of-the-art implementations [6].

Geosphere makes two key fundamental intellectual contributions to advance the state-of-the-art in MIMO wireless system design:

1. We observe that the sphere decoder can use the soft information about each received constellation point and the geometry of the constellation to prune the sphere decoder's search tree, in some cases without calculating the metrics associated with each node in the tree. This technique, which we term *geometrical pruning*, is analogous to the pruning step in the classical A* search algorithm [11], but is tailored for the particulars of the sphere decoder and the geometry of the transmitted constellation.
2. We introduce a new technique, *two-dimensional zigzag enumeration*, that approximates an expanding-ring search about each constellation point. This permits increasing the order of the transmitted constellation with only an incremental increase in processing requirements.

In the remainder, we begin with a primer on sphere decoding (§2), setting up our subsequent discussion of Geosphere's design (§3). A performance evaluation follows, where we measure Geosphere's performance in trace-driven simulation, using wireless traces from a 15-node wireless testbed built with Rice WARP version 3 radios [23]. Here we show that substantial throughput gains can be achieved in comparison with systems employing zero-forcing decoding, and that in spatial-division multiplexing systems where many single-antenna clients transmit at the same time, Geosphere increases the per-user throughput (*i.e.*, Geosphere achieves super-linear network throughput improvements). Results from our WARP testbed show that Geosphere achieves average throughput gains of $2\times$ in 4×4 MIMO systems and 47% in 2×2 MIMO systems, while simultaneously requiring up to nearly an order of magnitude less computation relative to the sphere decoder, bringing its computational demands in line with current systems already realized in ASIC. We survey related work in Section 6, and conclude in Section 7.

2 Primer: The Sphere Decoder

This section provides essential background on the sphere decoder [2, 8], an algorithm able to determine the most-likely transmitted bits $\mathbf{x}$ in the MIMO system that Equation 1 describes. Supposing transmitters send symbols chosen from a constellation $\mathcal{O}$ of size $|\mathcal{O}| = 2^Q$ (*i.e.*, Q bits per symbol), such a solution, called the *maximum-likelihood* solution, finds

$$\mathbf{x}^* = \arg \min_{\mathbf{s} \in \mathcal{O}^{n_c}} \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 . \tag{5}$$

This is the solution that minimizes the bit error rate, maximizing throughput. Unfortunately, the computational complexity of the exhaustive search in Equation 5 grows exponentially both in the message length and in the constellation size. For example, if we were to attempt to find the maximum-likelihood solution by exhaustive search, we would need to perform $|\mathcal{O}|^{n_c}$ Euclidean distance calculations. This means that for an OFDM system with 48 data sub-carriers, four antennas and a 4-QAM constellation, we would need to calculate approximately 10^4 Euclidean distances, but in the same system sending with 64-QAM, we would need approximately 10^9 distance calculations. Sphere decoding reduces this complexity while still likely finding the maximum-likelihood solution.

2.1 The sphere constraint

The sphere decoder constrains its search to only those possibilities $\mathbf{s}$ that lie within a hypersphere of radius r about the received vector $\mathbf{y}$, as measured by the *Euclidean distance* $d(\mathbf{s})$. This is the *sphere constraint*:

$$d(\mathbf{s}) < r^2, \text{where } d(\mathbf{s}) = \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 . \tag{6}$$

Most sphere decoders begin with $r \leftarrow \infty$, and upon discovering a solution at distance $r' < r$, safely reduce $r \leftarrow r'$, without the possibility of excluding the maximum-likelihood solution.

2.2 The tree

The sphere decoder recasts the maximum-likelihood problem (Equation 5) into a search in a tree of height n_c (number of client antennas) and branching factor $|\mathcal{O}|$ (constellation size). Figure 4 shows an example for $n_c = 3$ and QPSK ($|\mathcal{O}| = 4$) to which we will subsequently refer. Each level l of the tree corresponds to a decision on the value of the transmitted symbols from antennas l through n_c, which we will term a *partial symbol vector* $\mathbf{s}^{(l)} = [s_l, s_{l+1}, \ldots, s_{n_c}]$). Formulating the problem as a tree search requires the channel matrix $\mathbf{H}$ to be triangularized using a QR decomposition [27] into $\mathbf{H} = \mathbf{QR}$, where $\mathbf{Q}$ (of dimension $n_a \times n_c$) has the property that $\mathbf{Q}^*\mathbf{Q} = \mathbf{I}$ and $\mathbf{R} = [r_{ij}]$ (of dimension $n_c \times n_c$) is *upper-triangular* (*i.e.*, has zeroes below its diagonal). We can then rewrite the received signal (Equation 1) as

$$\widehat{\mathbf{y}} = \mathbf{Rs} + \mathbf{Q}^*\mathbf{w}, \text{where } \widehat{\mathbf{y}} = \mathbf{Q}^*\mathbf{y}, \tag{7}$$

and the Euclidean distances $d(\mathbf{s})$ as

$$d(\mathbf{s}) = K + \|\widehat{\mathbf{y}} - \mathbf{Rs}\|^2. \tag{8}$$

where K is an independent constant that can be safely ignored. Since $\mathbf{R}$ is upper-triangular, we can now calculate *partial Euclidean distances* for the partial symbol vectors, starting at the top of the tree at level n_c. We label each branch in the tree with a non-negative *branch cost*

$$c(\mathbf{s}^{(l)}) = \left| \widehat{y}_l - \sum_{j=l}^{n_c} r_{lj} s_j \right|^2. \tag{9}$$

As we walk down the tree from the root, a selected branch at level l prepends a new symbol s_l to the partial symbol vector $\mathbf{s}^{(l+1)}$, where $\mathbf{s}^{(l+1)}$ is the solution constructed up to the level above. We calculate the partial Euclidean distance for all $\mathbf{s}^{(l)}$ as

$$d(\mathbf{s}^{(l)}) = d(\mathbf{s}^{(l+1)}) + c(\mathbf{s}^{(i)}). \tag{10}$$

Since the branch cost is non-negative, the sphere decoder prunes all children below partial symbol $\mathbf{s}^{(l)}$ if

$$d(\mathbf{s}^{(l)}) \geq r^2, \tag{11}$$

as they will violate the sphere constraint. This pruning greatly reduces the number of solutions the sphere decoder needs to consider, but notice that further efficiencies are possible if we visit solutions closest to the maximum-likelihood solution earlier in our search. The efficiency of the sphere detector is thus to a large part determined by the tree-traversal strategy.

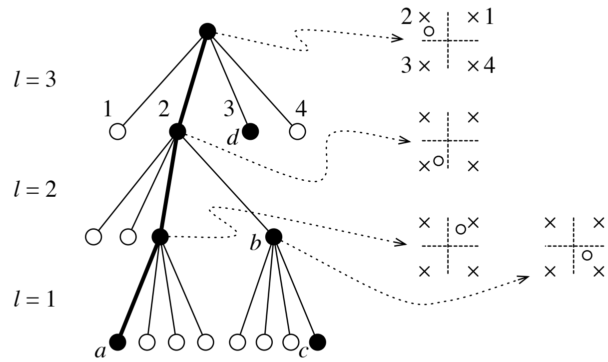

Figure 4: The sphere decoder operating on $n_c = 3$ transmit antennas, each sending a QPSK ($|\mathcal{O}| = 4$) symbol. Constellation points (denoted $\times$) and corresponding branches of the tree are numbered at the uppermost level ($l = 3$), and the received signal is denoted $\circ$. Visited nodes are colored black.

2.3 Traversing the tree

We begin with a depth-first tree-traversal strategy, as it is the approach we take in Geosphere for reasons that will become clear later. A simple yet powerful refinement of a textbook depth-first tree traversal is to visit children of a tree node in ascending order of their partial Euclidean distances, an idea known as *Schnorr-Euchner* enumeration [26] after its inventors.

Continuing our example of Figure 4, conventional Schnorr-Euchner sphere decoders will first greedily follow the path to a leaf a that minimizes partial Euclidean distance at each level (this path's branches are shown with thick lines in the figure). This entails computing distances for this path as well as all sibling nodes along the path (all nodes in this diagram). Upon reaching a, the decoder sets its sphere radius to $d(a)$ and backtracks up one level to check the node whose distance is second-closest, b. Let's assume that $d(b) < d(a)$; this means that the sphere decoder needs to expand b, search its children, and find the one with minimum distance (c). Once this is finished, the decoder backtracks up one level again to $l = 3$ and considers node d. Now $d(d) \geq d(a)$, so none of d's children could possibly be the maximum-likelihood solution, so the sphere decoder terminates and returns a as the maximum-likelihood solution.

It is clear that this pruning reduces the number of visited nodes, but reducing the number of visited nodes does not necessarily reduce processing requirements. In particular, the sorting requirement of Schnorr-Euchner enumeration is very computationally expensive for higher-order constellations (*e.g.*, 16- and 64-QAM), and can therefore compromise the sphere decoder's efficiency. In the foregoing example, in order to determine the node to visit we have fully enumerated and sorted all possibilities when we visited a node not violating the sphere constraint. This entails, at each step, calculating partial Euclidean distances for all possible children and then sorting them, a highly inefficient process, since we will spend processing power calculating distances for many nodes that we will never need to expand.

3 Design

This section presents the design of Geosphere, starting from the enumeration technique we use in order to efficiently sort children of a node in the sphere decoder (§3.1), and continuing to describe the improved pruning technique that we propose (§3.2). Later in Section 5, we will experimentally evaluate the relative gains of each to highlight the different roles the two techniques play when channel conditions are poor and favorable.

3.1 Constellation point enumeration

The goal of Geosphere's enumeration technique is to determine the order that the sphere decoder should explore the set of constellation points $\mathcal{O}$, when it is considering which branch to expand at a particular node in the sphere decoding tree shown in Figure 4 on the preceding page. We wish to explore constellation points in order of increasing branch cost, but the only soft information at our disposal is the received symbol.

However, since constellation distance is related to partial Euclidean distance by

$$c\left(\mathbf{s}^{(l)}\right) = |r_{ll}|^2 \left|\widetilde{\mathbf{y}}_l - s_l\right|^2 \tag{12}$$

(where $\widetilde{\mathbf{y}}_l = \frac{\widehat{y}_l - \sum_{j=l+1}^{n_c} r_{lj} s_j}{r_{ll}}$), it suffices to explore the constellation points in increasing Euclidean distance from the received symbol in the constellation itself, rather than as measured indirectly by the partial Euclidean distance metric.

If we were sending constellation points in one dimension (this is known as *pulse-amplitude modulation*, or *PAM*), the task is substantially easier, so we discuss this case first. Figure 5 *(left)* shows a PAM constellation comprised of four constellation points ($\times$) and a received symbol ($\circ$). To find the closest constellation point to the received symbol we compare the received symbol against the decision boundaries indicated by the vertical dotted lines in the figure (this procedure is called *slicing* the received symbol), and therefore order constellation point *(a)* first. The zigzag rule tells us to visit the next closest, unvisited constellation point from *(a)* in the direction of the received symbol; this is *(b)* in the figure. Subsequent applications of the same rule take us to *(c)* and then *(d)*.

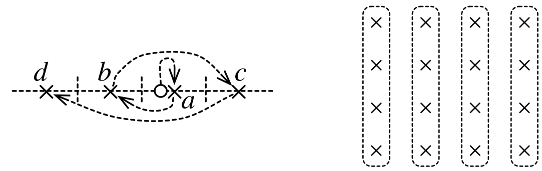

Figure 5: *Left:* The zigzag technique in a one-dimensional (PAM) constellation visits constellation points ($\times$) in increasing distance from the received symbol ($\circ$). *Right:* Dividing a 16-QAM constellation into four 4-PAM subconstellations.

3.1.1 Two-dimensional zigzag enumeration

Now let's consider the two-dimensional case. We are in fact seeking an approximation of an expanding ring search, starting at an arbitrary, continuous-valued received symbol point $\circ$. One inexact way of accomplishing this would be to partition the QAM constellation into PAM subconstellations as shown in Figure 5 *(left)*, and then zigzag "vertically" within each subconstellation. But this approach neglects the *in-phase* (*i.e.*, horizontal, abbreviated *I*) component of the received symbol.

So instead Geosphere first slices the received symbol to find the closest constellation point (call it a), and begins the two-dimensional zigzag from that exact constellation point. Note that the sphere decoder will then expand the branch corresponding to a and search that subtree. Once the sphere decoder returns to the node whose constellation points we are sorting, should we zigzag horizontally or vertically? We try both, since we are trying to find the next-closest constellation point in (two-dimensional) Euclidean distance, with the exception that we avoid a horizontal zigzag if a constellation point from the target PAM subconstellation is already in our list of outstanding constellation points to explore. This ensures that we have at most one candidate constellation point per (vertical) PAM subconstellation.

Figure 6 on the next page shows the pseudocode for the algorithm. Notice that as a consequence of the two-dimensional zigzag rule, the algorithm needs a priority queue of length at most $\sqrt{|\mathcal{O}|}$. By only taking zigzag steps one constellation point at a time, the algorithm defers the Euclidean distance computation until as late as possible, often by which time the sphere decoder has pruned the relevant subtree. We demonstrate this in the experimental evaluation following this section.

Example. Figure 7 on the following page shows an example of the two-dimensional zigzag algorithm working in a 16-QAM constellation. In each frame, we show the 16-QAM constellation points ($\times$) alongside the received symbol ($\circ$), above the priority queue Q. In Step *(i)*, the

Algorithm (two-dimensional zigzag)

1. Initialize a sorted priority queue $Q = \emptyset$, comprising constellation points (maintain Q sorted by Euclidean distance to $\circ$ at all times).
2. Find the closest constellation point a to the received symbol by slicing $\circ$ on the constellation's decision boundaries. constellation's decision boundaries). Calculate a's Euclidean distance and enqueue $a \to Q$.
3. Dequeue $Q \to x$ and explore x's children in the sphere decoder.
   (a) Zigzag vertically from x with respect to $\circ$: call the result z_v. Calculate z_v's Euclidean distance to $\circ$ and enqueue $z_v \to Q$.
   (b) Zigzag horizontally from x with respect to $\circ$; call the result z_h. If no other constellation point in z_h's PAM subconstellation is in Q, calculate z_h's Euclidean distance to $\circ$ and enqueue $z_h \to Q$.
4. Go to Step 3.

Figure 6: Pseudocode for Geosphere's two-dimensional zigzag algorithm.

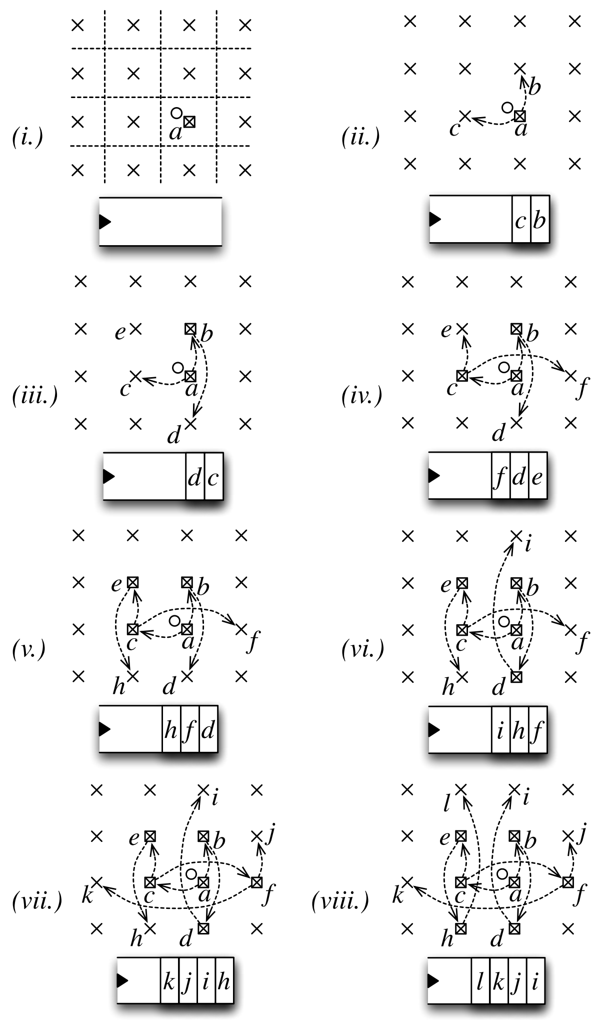

Figure 7: Geosphere's two-dimensional zigzag enumeration in the 16-QAM constellation. We denote constellation points ×, label constellation points whose partial Euclidean distances have been computed, and denote constellation points that have been explored ⊠.

slicer finds the closest constellation point to the received symbol, a. The sphere decoder explores a, zigzags vertically and horizontally, and enqueues b and c, respectively in Step *(ii)*. Since b is closer of b and c to $\circ$, in Step *(iii)* the algorithm explores and zigzags from b. But notice that a horizontal zigzag step from b to e would land in the same PAM subconstellation as a previously-explored constellation point (c). Consequently, we only zigzag vertically from b, enqueuing d. In Step *(iv)*, we explore and zigzag from c, picking up e and visiting all four constellation points surrounding $\circ$ (the closest to $\circ$) in Step *(v)*. Subsequent steps continue in the same manner, filling in the "expanding ring" around a, b, c, and e.

3.2 Geometrical pruning

We now turn to Geosphere's approach to pruning off whole sections of the sphere decoder's search tree, a key step in making the search process tractable in practice.

Suppose that the sphere decoder has identified a currently-best candidate node a (referring to Figure 8) somewhere in the tree, and now keeps track of the associated partial Euclidean distance $d(a)$. Recall from Section 2 that when the sphere decoder visits node x elsewhere in the tree it considers whether or not to prune each branch emanating from x. The most straightforward way of doing this is to evaluate the exact branch cost $c\left(\mathbf{s}^l\right)$ for each, but this requires two floating-point multiplication operations and an addition.

Geosphere instead uses the constellation's geometry

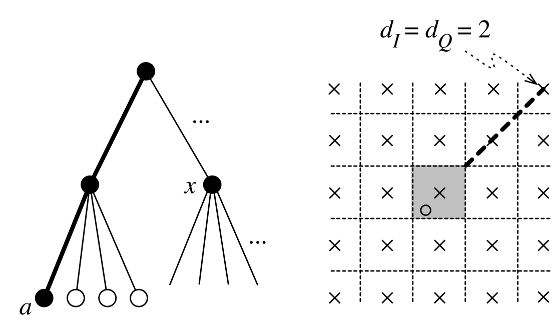

Figure 8: Geometrically lower-bounding the distance between received symbol $\circ$ and another constellation point at horizontal (d_I) and vertical (d_Q) offset two from the closest constellation point. Constellation points are spaced two units apart.

to establish a lower-bound on the exact branch cost, as shown in Figure 8 on the preceding page. If the constellation point corresponding to the branch being tested is offset from the nearest constellation point by d_I horizontally and d_Q vertically, Geosphere computes

$$\hat{c}\left(\mathbf{s}^{l}\right) = \sqrt{\left(2d_I - 1\right)^2 + \left(2d_Q - 1\right)^2} \qquad (13)$$

based on a fast table lookup indexed on $|d_I|$ and $|d_Q|$ instead of floating-point operations, and uses $\hat{c}(\cdot)$ instead of $c(\cdot)$ in its pruning decision. Since $\hat{c}(\cdot) \leq c(\cdot)$, pruning based on Equation (13) alone doesn't exclude the maximum-likelihood solution, but may expand more branches than pruning based on exact branch costs. For this reason, if the above geometrical pruning test fails to exclude a branch, we calculate the branch's exact cost and attempt to exclude the branch on that basis.

4 Implementation

We implement Geosphere on Rice WARP v3 radio hardware and WARPLab software. Using WARPLab, we implement OFDM modulation and demodulation using 4-, 16- and 64-QAM constellations. All clients send data using 1/2-rate convolutional coding (similar to recent 802.11 standards), and transmitted packets are limited in size to 500 Kbytes due to restrictions on the maximum packet size that WARPLab can handle.

5 Evaluation

In this section we measure Geosphere's throughput performance gains and complexity requirements in real indoor office conditions. First, we show that the indoor wireless channel is often quite poorly-conditioned, and that zero forcing-based techniques are leaving performance on the table. Then, in terms of throughput, we compare Geosphere with MIMO systems that use zero-forcing and attempt to intelligently adapt to poorly-conditioned MIMO channels by varying the number of antennas and spatial streams they use. Finally, we evaluate the computational complexity of Geosphere, comparing it with the well-established depth-first sphere decoder of [12] which is one the few, and probably the most efficient solution able to provide the exact maximum-likelihood performance. Table 1 on the following page summarizes the experimental results presented here.

Testbed setup. Our testbed consists of single-antenna clients and four-antenna APs, communicating over a wireless channel of 20 MHz bandwidth in the 5 GHz ISM band. The distance between consecutive AP antennas is about 20 cm (approximately 3.2λ, where λ is the wireless

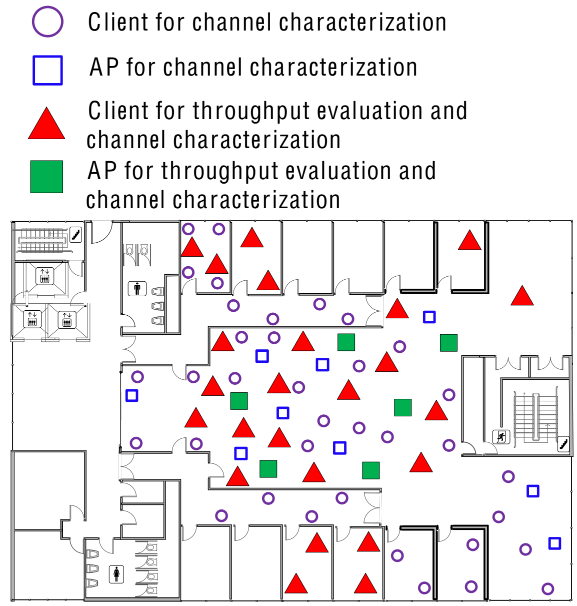

Figure 9: Floor plan of the office space housing the wireless testbed used in Geosphere's experimental evaluation.

wavelength) so that the wireless channels from each AP antenna to a client are uncorrelated with each other.

We evaluate Geosphere in actual office conditions: Figure 9 shows the testbed environment, including the places we position APs and clients. Note that the topology includes both line-of-site and non-line-of-site paths due to furniture and people but also due to transmissions penetrating through and reflecting off walls.

5.1 Channel characterization

As we mention above (§1), when the channel is well-conditioned, zero-forcing is a very efficient way to demultiplex the interfering streams. So, the question that naturally arises is whether the channels we face in an indoor environment are typically well-conditioned. Or equivalently, is there throughput on the table for Geosphere?

Methodology. To answer this question we measure the corresponding MIMO channels, for several concurrently-transmitted streams, across all OFDM subcarriers (*i.e.* on slightly different frequencies), and for many different positions of the clients and APs. Figure 9 shows their exact positions, with hollow circles and red triangles denoting client positions in this experiment, and squares denoting APs.

In order to characterize the corresponding channels we will use two metrics. The first is the square of the condition number $\kappa^2(\mathbf{H})$ which as discussed in 1 is a good

Experiment	Section	Conclusion
Channel characterization	§5.1	2×2 indoor MIMO channels are poorly-conditioned 60% of the time; 4×4 indoor MIMO channels are almost always poorly-conditioned.
Throughput comparison	§5.2	Geosphere achieves $2\times$ throughput gains over multi-user MIMO for four AP antennas and four clients, and 47% throughput gains over multi-user MIMO for the 2×2 case.
Computational complexity	§5.3	Geosphere reduces the required computation for the sphere decoder by nearly one order of magnitude over the ETH-SD sphere decoder ([12], *cf.* §5.3), making the sphere decoder practical for dense constellations.

Table 1: A summary of the major experimental results in this paper.

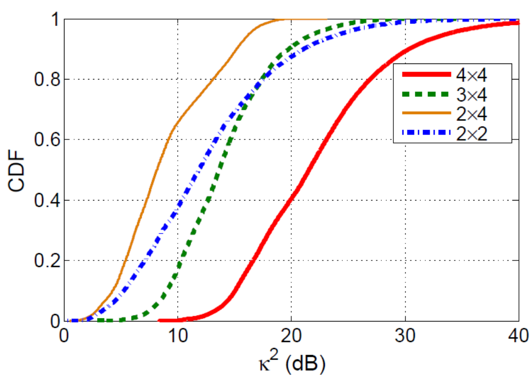

Figure 10: Cumulative distribution across testbed links, OFDM subcarriers, and spatial streams of κ^2 (decibels), the power of the MIMO channel condition number. Higher values of κ^2 indicate worse channel conditioning.

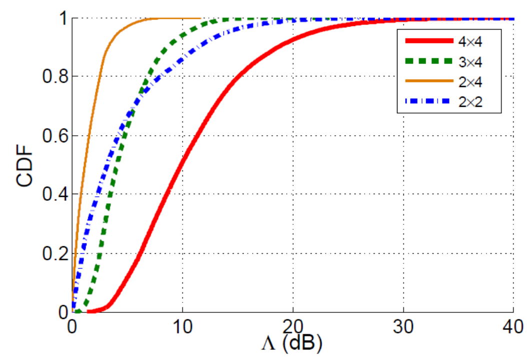

Figure 11: Cumulative distribution across testbed links and OFDM subcarriers of Λ, the signal-to-noise ratio degradation that the most-degraded user experiences for each particular link.

upper-bound on the actual noise amplification due to zero-forcing. However, this metric doesn't necessarily provide us with the actual performance difference between a zero-forcing system and one using a maximum-likelihood detector like Geosphere.

The metric of most interest is the signal-to-noise ratio (SNR) of the k^{th} transmitted stream after being transmitted over the MIMO channel $\mathbf{H}$: $\frac{[\mathbf{H}^*\mathbf{H}]_{k,k}}{2\sigma^2}$. The SNR of the same stream after zero-forcing can be easily calculated to be $\frac{1}{[(\mathbf{H}^*\mathbf{H})^{-1}]_{k,k}2\sigma^2}$. Thus, the SNR degradation for stream k is $\lambda_k = \frac{[\mathbf{H}^*\mathbf{H}]_{k,k}}{[(\mathbf{H}^*\mathbf{H})^{-1}]_{k,k}}$.

We are interested in limiting the damage done to any particular user, and so we define our figure of merit Λ to be the maximum over λ_k. In other words, Λ denotes the worst (over clients) SNR degradation due to zero-forcing noise amplification.

Results. In Figure 10 and Figure 11 we show the the cumulative distributions of κ^2 and Λ respectively, partitioned by different numbers of clients and receive antennas at the AP. In the two-client, two receive antenna case (*i.e.*, 2×2), 60% of the links experience channels with condition numbers larger than 10 dB while in the 4×4 case, nearly all links are poorly conditioned.

To characterize the links in terms of the maximum SNR degradation that any particular user sees we refer to the cumulative distributions of Λ (Figure 11). We observe that the use of zero-forcing will result in 30% of the MIMO channels experiencing an SNR degradation of more than 5 dB, while 90% of the channels will face such a degradation for 4×4 links. This shows that there are a significant number of situations where Geosphere can substantially increase throughput, especially when simultaneously serving more clients (increasing both the clients and receive antennas) as systems such as BigStation [32] do. From Figures 10 and 11, we also see that if we fix the number of receive antennas to a large number (*e.g.* four), we can achieve a better-conditioned channel by decreasing the number of clients transmitting simultaneously. For example, if only two clients transmit, the maximum degradation to due to zero-forcing will be less than three decibels for 90% of the channels. That means that we can sacrifice channel capacity to reduce the degradation due to zero-forcing. Then the question is, if we do that, do we still need Geosphere? This is one of the questions we will answer next.

5.2 System throughput

We now examine the uplink throughput that zero-forcing achieves when serving a network of clients, in comparison with Geosphere. The previous discussion shows that due to characteristics of the channel there is an opportunity for throughput improvement for Geosphere, especially for the 4×4 case; we now examine whether Geosphere can realize these gains in practice.

Methodology. We position clients and APs in a subset of the positions used for channel measurements, denoted by hollow circles and hollow squares respectively in Figure 9 on page 7. We send data to the AP using various modulations to characterize performance at different sending rates: we transmit all 4-, 16- and 64-QAM constellations. We note that the channel is changing due to people walking nearby. We also note that for this subset of positions the condition number and the Λ values of the links are smaller than those when all positions are included. Therefore, we are evaluating here a particularly challenging case for Geosphere.

We consider three SNR ranges, 15 dB ± 5 dB, 20 dB $\pm$ 5 dB, and 25 dB ± 5 dB, where the SNR in question is the average SNR over all transmitted streams.

Results. In lieu of implementing an rate adaptation algorithm, we show throughput results for the constellation that achieves the best average throughput for the corresponding range; this emulates an ideal bit rate adaptation algorithm. In Figure 12 on the next page we show achieved throughput for different numbers of clients and receive antennas. We can see that Geosphere consistently provides better throughput than zero-forcing. Moreover, as expected, the throughput gains increase with the condition number and Λ. In particular, for the 2×2 case Geosphere can provide a throughput increase of up to 47%, while for the 4×4 case it can be more than two times faster. Even in the worst case for Geosphere of two and three clients and an AP with receive antennas where the corresponding channels are most often well-conditioned, Geosphere provides gains of 6%. These throughput gains are consistent with what we where expecting from our channel characterization.

Since the condition number of a matrix becomes smaller with decreasing numbers of concurrently transmitting clients, another question we may ask is whether zero-forcing and an appropriate time-division scheduling strategy could equal Geosphere's performance, with fewer clients per timeslot. But Figure 12 on the following page shows that this is not in fact true. Geosphere with four clients and four receive antennas consistently provides better performance than a zero-forcing scheme which three transmitting clients, with throughput gains that can be up to 36% (at 20 dB).

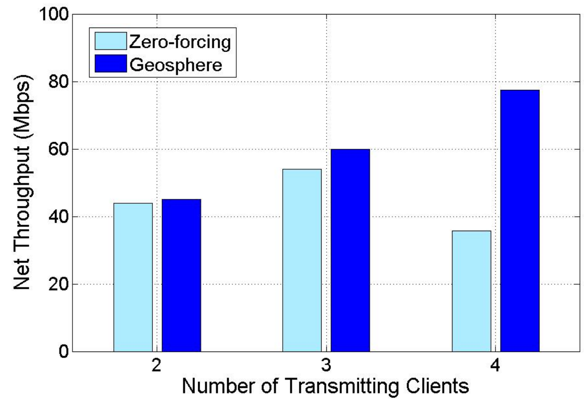

Figure 13: Throughput comparison between zero-forcing MIMO and Geosphere for different numbers of users accessing a four-antenna AP at the same time.

Figure 13 shows the achievable uplink throughput of zero-forcing and Geosphere for a four-antenna AP when we increase the number of clients at 20 dB. We see Geosphere achieves linear gains in throughput with the number of clients while zero-forcing does not.

5.3 Computational complexity

We compare Geosphere against the most efficient known depth-first sphere decoder implementation able to achieve the maximum-likelihood solution (we denote this system *ETH-SD* in the following experimental results). In particular we base our implementation of ETH-SD on the sphere decoding implementation of [6] but instead of decomposing the constellation to equivalent constant-amplitude sub-constellations (*i.e.*, PSK ones) we use the superior method of [12] which splits the QAM constellation to horizontal sub-constellations, performs one-dimensional zigzag and compares Euclidean distances across all sub-constellations to determine the node we will visit next. This approach is more efficient for dense constellations since it involves fewer sub-constellations.

Methodology. First we compare the complexity of Geosphere and ETH-SD using the real experiments we have collected. In particular, we decode the collected traces using both sphere decoders and we compare their complexity in terms of partial distance calculations since these are the main sources of complexity. Since in an OFDM

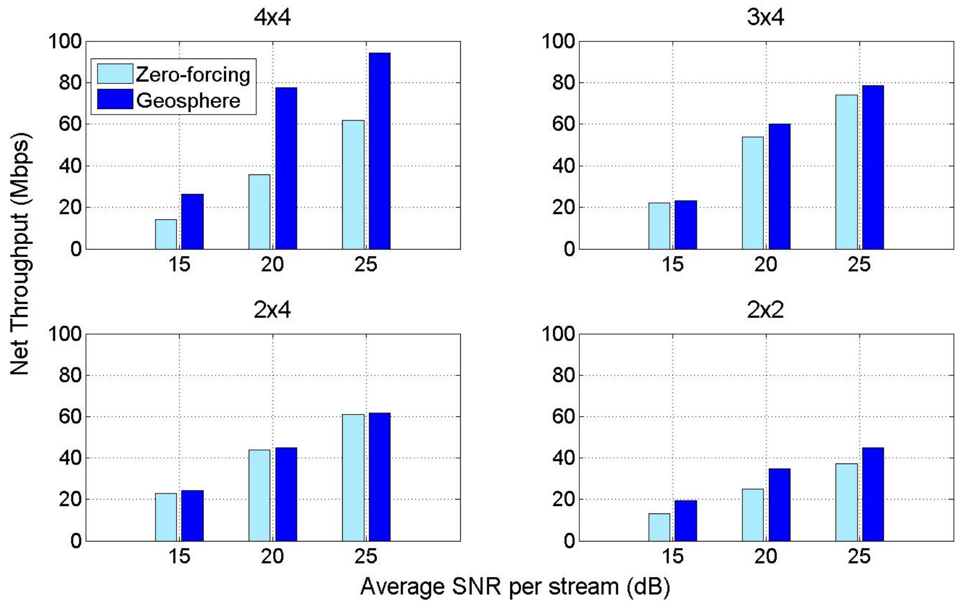

Figure 12: Throughput comparison between zero-forcing MIMO and Geosphere for different numbers of clients and AP antennas.

system the MIMO processing takes place per sub-carrier, we show the average required partial distance calculations across data sub-carriers. In order to show that Geosphere is the first (to our best knowledge) sphere decoder appropriate for the detection of very dense constellations, and since our WARP platforms cannot reach the required SNR, we perform trace-driven simulations, using the transmission channels collected from our live testbed experiments.

Results. In Figure Figure 14 on the following page we show the average number of partial distance calculations for all experiments. We see that Geosphere is consistently less complex than ETH-SD, and the gains increase when SNR increases, due to fact that Geosphere is more efficient in dense constellations. In the 25 dB range, our complexity gains can be up to 63%.

As we discussed in the previous paragraphs, the throughput gains of Geosphere are modest for well-conditioned channels. So, one could argue that our approach is not needed, and we ought to switch from Geosphere to zero-forcing. However, the above results show that Geosphere adjusts its computational complexity to the transmission environment, and so its complexity for well-conditioned matrices is actually very small.

In Figure 15 we perform simulations to see the complexity of Geosphere for dense constellations and we split the gains of Geosphere to zigzag only and full. We show complexity for the SNR such that each constellation can reach a frame error rate of 10%. We see that while the complexity of ETH-SD greatly increases with the order of constellation, this is not the case for Geosphere. As a result Geosphere is about an order of magnitude less complex than ETH-SD. In addition we see that the zigzag algorithm is the main source of complexity improvement for large constellations while the early pruning is significant for lower constellation sizes.

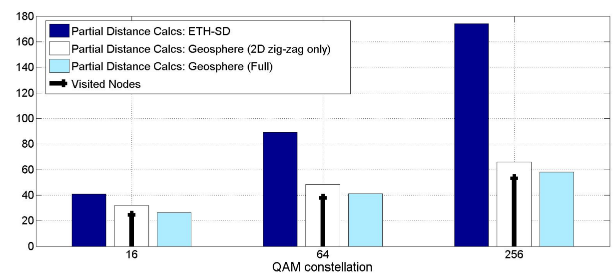

Figure 15: Complexity comparison between zero-forcing MIMO and Geosphere for different numbers for dense QAM constellations through simulations.

6 Related Work

Work on the sphere decoder has been extensive, and we are not the first to note the importance of and measure the condition number of the MIMO channel, and propose solutions for noise amplification. We now discuss related work in these two areas, as well as in their intersection, placing Geosphere into context and highlighting our contributions. We finish the section with prior work relevant

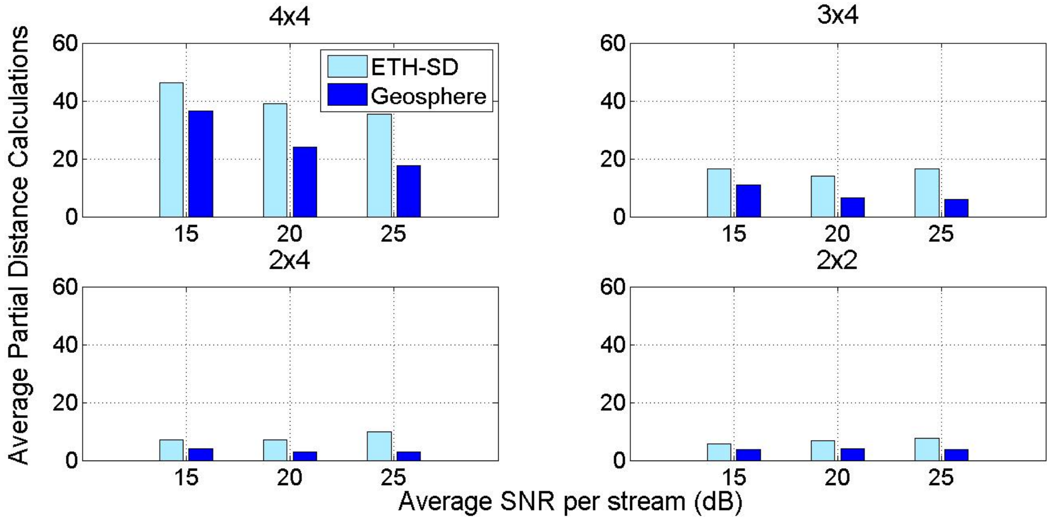

Figure 14: Complexity comparison between zero-forcing MIMO and Geosphere for different numbers of clients and AP antennas.

to wireless constellation geometry.

Sphere decoder optimizations. Other sorting approaches that approximate Euclidean distance with simpler norms have been proposed to mitigate sphere decoder processing overhead while preserving maximum-likelihood optimality [6, 12]. However, their computational overhead remains high when sending dense constellations, making them impractical at high data communication rates.

Zhao and Giannakis [33] generalize Schnorr-Euchner enumeration probabilistically to reduce sphere decoder complexity, but by their own admission, their techniques are only beneficial in the high-SNR regime ($>$ 22 dB). By comparison, Geosphere's techniques are effective over the entire range of SNRs commonly found in wireless local-area networks (see §5).

Another body of work, *e.g.* Peel *et al.* [21], precodes (*i.e.* alters, across antennas) information at the transmitter in order to simplify the problem. Precoding, however, requires that clients track the wireless channel as they move, which adds complexity and can add overhead to the system. Nonetheless, Geosphere is complementary to precoding: we expect the two should achieve complementary performance gains if implemented together.

Breadth-first sphere decoders. In this work we have focused our discussion on depth-first sphere decoders, as Geosphere takes this approach, but there is a large body of work on sphere decoders that explore the decoding tree breadth-first instead.

The fixed-complexity sphere decoder [4] is a specific type of breadth-first sphere decoder that initially searches the first p levels of the tree, then plunges depth first, but using a branching factor of only one. Jaldén *et al.* show that the fixed-complexity sphere decoder can only asymptotically reach maxium-likelihood performance at high SNRs [16]. We view the fixed-complexity sphere decoder work as complementary to Geosphere, as the key zigzag and geometrical pruning techniques that we propose in this work can also be applied to breadth-first sphere decoders; we leave an exploration of this for future work.

Finally, we note that the Spinal codes [22] decoder resembles a breadth-first sphere decoder with a bounded branching factor at each level. However, Spinal codes uses a novel encoder design that improves performance. With regards to Geosphere, Spinal Codes are designed for a point-to-point wireless channel, not the multi-antenna MIMO channel, but we speculate that they may be extended to the MIMO channel in the future.

Channel conditioning and noise amplification. While the MIMO channel condition number has been previously measured, published measurements are mostly associated with mobile cellular systems, and thus often taken outdoors (*e.g.*, Teague *et al.* [29], in the 2.16–2.18 GHz frequency band), indoors, but in a mobile celluar frequency band (*e.g.*, Kita *et al.* [17]), or in an unspecified environment (*e.g.*, Agilent Corp. [1]). Nonetheless, we note that the MIMO channel condition number distributions obtained in this related work are roughly comprable to our measurements (§1, §5), suggesting that the problem of poor channel conditioning occurs in general, in outdoor as well as indoor environment, and across the range of microwave communication carrier frequencies.

Channel hardening [13, 15] refers to the linear increase in throughput possible in zero-forcing multi-user MIMO systems, when the number of antennas increases dramatically. This is due to the ability of the access point to select a set of antennas that results in a well-conditioned MIMO channel matrix. Among its results, this theoretical work shows that many more antennas than users are required to acheive linear throughput gains.

The minimum mean-squared error (MMSE) detector is an improvement on zero-forcing of similar complexity that balances between completely decoupling the interfering streams and amplifying noise. However, MMSE cannot provide substantial throughput gains compared to zero-forcing in the medium and the high signal-to-noise ratio regime [31].

In recent wireless systems work, the authors of BigStation [32] have speculated that their zero-forcing multi-user MIMO access point may require more than 40 antennas (or $2\times$ the number of users) in order to mitigate the problem of a MIMO channel hardening. In this context, our work on Geosphere offers an alternative solution to dramatically increasing the numbers of antennas and radios (with their associated costs) at the access point.

Chen and Wang [7] analyze the interaction of zero-forcing and time-division scheduling techniques, deriving closed-form analytical throughput expressions.

Channel condition-aware sphere decoders. These sphere decoders adapt their behavior (or even switch between zero-forcing and sphere decoding) based on the channel condition number $\kappa(\mathbf{H})$.

With measurements from random, simulated MIMO channels, Artés *et al.* [3] also note the effect of the condition number on the zero-forcing decoder, and propose a linear filter that compensates for the distortion the zero-forcing decoder introduces. While this method makes few bit errors a small constellation size (*i.e.*, four), it has not been shown to scale to larger constellations Leveraging the power of the sphere decoder, Geosphere maintains performance while scaling to 256-QAM.

Sayana *et al.* [25] use successive interference cancellation [31] and soft information to reduce the effects of noise amplification in a MIMO system, but their design is tied to a specific type of coding and modulation (bit-interleaved coded modulation), whereas Geosphere is generalizable to many different coding schemes.

Maurer *et al.* propose a system that switches between zero-forcing and maximum-likelihood decoding via a threshold test on the channel condition number [20]. However, unlike Geosphere, they do not present experimental results with a real sphere decoder, and use random matrices rather than real MIMO wireless channel matrices, calling into question the practical applicability of their simulation-based results. Also missing is a means of choosing the switching threshold. In a similar vein, Roger *et al.* [24] propose a sphere decoder that expands at most K branches of each node in the decoding tree, varying K based on $\kappa(\mathbf{H})$. Compared to both works, Geosphere makes the sphere decoder practical with new techniques that markedly reduce its complexity, and we present a full working system design and experimental evaluation in real indoor office conditions.

Constellation geometry-aware approaches Brown *et al.* [5] use log-likelihood ratios to compute decision regions for the various bits of a grey-coded constellation. Geosphere's two-dimensional zigzag enumeration can be extended to use these decision boundaries to reduce the number of distance computations even further; we leave this optimization for future work.

7 Conclusions and Future Work

We have described Geosphere, a wireless multi-user MIMO system that consistently achieves higher uplink throughputs than similar systems based on zero-forcing. Geosphere makes the sphere decoder practical in a real wireless system sending at high rates (using dense constellations) with two new ideas: two-dimensional zigzag enumeration, and geometrical pruning.

Scope. We note that since sphere decoder-based techniques need to exchange information between receiving antennas, Geosphere provides performance improvements only in the case of the uplink where many clients are transmitting to a single AP, or over individual links where one AP or client with many antennas transmits (uplink or downlink) to another AP or client with many antennas. However, with the shifting ratio of downlink to uplink traffic driven by file system backup, VOIP, and video telephony, performance improvements in the uplink are direly needed and increase overall spectrum efficiency.

As a consequence of using the sphere decoder, Geosphere is extendable to iterative *soft-input, soft-output* sphere decoders, which combine error control coding with MIMO to achieve theoretical rates very close to the MIMO channel capacity. Since these soft-input, soft-output sphere decoders can be decomposed to many conventional sphere decoders running in parallel [14], we speculate that we may be able to leverage the techniques proposed in this work in soft-input, soft-output designs. One outstanding set of challenges here is how to manage the corresponding complexity while meeting latency and power consumption requirements.

References

[1] Agilent Technologies. MIMO Performance and Condition Number in LTE Test: Application Note, 2009.

[2] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger. Closest point search in lattices. *IEEE Trans. Inf. Theory*, 48(8):2201–2214, Aug. 2002.

[3] H. Artés. Efficient detection algorithms for MIMO channels: a geometrical approach to approximate ML detection. *IEEE Trans. on Signal Proc.*, 51(11):2808–2820, Nov. 2003.

[4] L. Barbero and J. Thompson. Fixing the complexity of the sphere decoder for MIMO detection. *IEEE Trans. on Wireless Comms.*, 7(6):2131–2142.

[5] J. Brown, S. Pasupathy, and K. Plataniotis. Adaptive demodulation using rateless erasure codes. *IEEE Trans. on Comms.*, 54(9):1574–1585, Sept. 2006.

[6] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bolcskei. VLSI implementation of mimo detection using the sphere decoding algorithm. *Solid-State Circuits, IEEE Journal of*, 40(7):1566–1577, 2005.

[7] C.-J. Chen and L.-C. Wang. On the performance of the zero-forcing receiver operating in the multiuser MIMO system with reduced noise enhancement effect. In *Proc. of IEEE Globecom*, 2005.

[8] M. Damen, H. El Gamal, and G. Caire. On maximum-likelihood detection and the search for the closest lattice point. *IEEE Trans. Inf. Theory*, 49(10):2389–2402, Oct. 2003.

[9] S. Gollakota, S. Perli, and D. Katabi. Interference alignment and cancellation. In *Proc. of ACM SIGCOMM*, Aug. 2009.

[10] A. Gudipati and S. Katti. Strider: Automatic rate adaptation and collision handling. In *Proc. of ACM SIGCOMM*, 2011.

[11] P. Hart, N. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[12] C. Hess, M. Wenk, A. Burg, P. Luethi, C. Studer, N. Felber, and W. Fichtner. Reduced-complexity MIMO detector with close-to ML error rate performance. In *ACM Great Lakes Symp. on VLSI*, 2008.

[13] B. Hochwald, T. Marzetta, and V. Tarokh. Multiple-antenna channel hardening and its implications for rate feedback and scheduling. *IEEE Trans. on Info. Theory*, 50(9), Sept. 2004.

[14] B. Hochwald and S. Ten Brink. Achieving near-capacity on a multiple-antenna channel. *IEEE Trans. Commun.*, 51(3):389–399, Mar. 2003.

[15] B. Hochwald and S. Vishwanath. Space-time multiple access: Linear growth in the sum rate. In *Proc. of the Allerton Conf. on Comms., Control, and Computing*, 2002.

[16] J. Jaldén, L. Barbero, B. Ottersten, and J. Thompson. The error probability of the fixed-complexity sphere decoder. *IEEE Trans. on Signal Proc.*, 57(7):2711–2720, 2009.

[17] N. Kita, W. Yamada, A. Sato, D. Mori, and S. Uwano. Measurement of Demel condition number for 2x2 MIMO-OFDM broadband channels. In *Proc. of IEEE VTC*, 2004.

[18] E. Kreyszig. *Advanced Engineering Mathematics*. Wiley & Sons, Inc., 2006.

[19] K. Lin, S. Gollakota, and D. Katabi. Random access heterogeneous MIMO networks. In *Proc. of ACM SIGCOMM*, Aug. 2011.

[20] J. Maurer, G. Matz, and D. Seethaler. Low-complexity and full-diversity MIMO detection based on condition number thresholding. In *Proc. of the IEEE ICASSP Conf.*, 2007.

[21] C. Peel, B. Hochwald, and A. Swindlehurst. A vector-perturbation technique for near-capacity multiantenna multiuser communication—Part I: Channel inversion and regularization. *IEEE Trans. on Comms.*, 53(1):195–202, Jan. 2005.

[22] J. Perry, P. Iannucci, K. Fleming, H. Balakrishnan, and D. Shah. Spinal codes. In *Proc. of ACM SIGCOMM*, 2012.

[23] Rice Univ. Wireless Open Access Research Platform (WARP). http://warp.rice.edu/trac.

[24] S. Roger, A. Gonzalez, V. Almenar, and A. Vidal. Combined K-best sphere decoder based on the channel matrix condition number. In *Proc. of the IEEE ISCCSP Symp.*, 2008.

[25] K. Sayana, S. Nagaraj, and S. Gelfand. A MIMO zero-forcing receiver with soft interference cancellation for BICM. In *Proc. of the IEEE Workshop on Sig. Proc. Advances in Wireless Comms.*, 2005.

[26] C. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Programming*, 66(2):181–191, Sept. 1994.

[27] G. Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 4^{th} edition, 2009.

[28] K. Tan, H. Liu, J. Fang, W. Wang, J. Zhang, M. Chen, and G. Voelker. SAM: Enabling practical spatial multiple access in wireless LAN. In *Proc. of ACM MobiCom*, 2009.

[29] H. Teague, C. Patel, D. Gore, H. Sampath, A. Naguib, T. Kadous, A. Gorokhov, and A. Agrawal. Field results on MIMO performance in UMB systems. In *Proc. of IEEE VTC*, 2008.

[30] I. Telatar. Capacity of multi-antenna Gaussian channels. *Eur. Trans. Telecomms.*, 10(6):585–596, Dec. 1999.

[31] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[32] Q. Yang, X. Li, H. Yao, J. Fang, K. Tan, W. Hu, J. Zhang, and Y. Zhang. BigStation: Enabling scalable real-time signal processing in large MU-MIMO systems. In *Proc. of ACM SIGCOMM*, 2013.

[33] W. Zhao and G. Giannakis. Reduced complexity closest point decoding algorithms for random lattices. *IEEE Trans. on Wireless Comms.*, 5(1):101–111, 2006.