Geosphere: Consistently Turning MIMO Capacity into Throughput

Konstantinos Nikitopoulos[†] Juan Zhou[‡] Ben Congdon[‡] Kyle Jamieson[‡]

[†]5G Innovation Centre
University of Surrey
k.nikitopoulos@surrey.ac.uk

[‡]Department of Computer Science
University College London
{juan.zhou.12, ben.congdon.11, k.jamieson}@ucl.ac.uk

ABSTRACT

This paper presents the design and implementation of Geosphere, a physical- and link-layer design for access point-based MIMO wireless 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 data streams by mathematically inverting the wireless channel matrix. In general, zero-forcing is highly effective, significantly improving throughput. But in certain physical situations, the MIMO channel matrix can become "poorly conditioned," harming performance. With these situations in mind, Geosphere uses sphere decoding, a more computationally demanding technique that can achieve higher throughput in such channels. To overcome the sphere decoder's computational complexity when sending dense wireless constellations at a high rate, Geosphere introduces search and pruning techniques that incorporate novel geometric reasoning about the wireless constellation. These techniques reduce computational complexity of 256-QAM systems by almost one order of magnitude, bringing computational demands in line with current 16- and 64-QAM systems already realized in ASIC. Geosphere thus makes the sphere decoder practical for the first time in a 4×4 MIMO, 256-QAM system. Results from our WARP testbed show that Geosphere achieves throughput gains over multi-user MIMO of $2\times$ in 4×4 systems and 47% in 2×2 MIMO systems.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication

Keywords

MIMO; Distributed MIMO; Sphere Decoder

1. INTRODUCTION

One of the most important challenges in modern wireless networks is to meet many users' increasing demands for throughput. One example from the workplace or home is when tens of users are running video telephony sessions, resulting in a high demand for symmetric (uplink and downlink) bandwidth. One way of meeting the demand for more throughput is by a technique called *spatial multiplexing*. Networks that leverage spatial multiplexing increase capacity [58] and throughput by sending multiple data streams from different transmit antennas. If enough receiving antennas hear the resulting mixture of signals and channel conditions are favorable, such systems can deliver the 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 wireless networks today. The throughput possible in these networks is determined, and thus constrained by, the number of available antennas at the access point (AP).

SIGCOMM'14, August 17–22, 2014, Chicago, Illinois, USA.

ACM 978-1-4503-2836-4/14/08 ...$15.00.
http://dx.doi.org/10.1145/2619239.2626301.

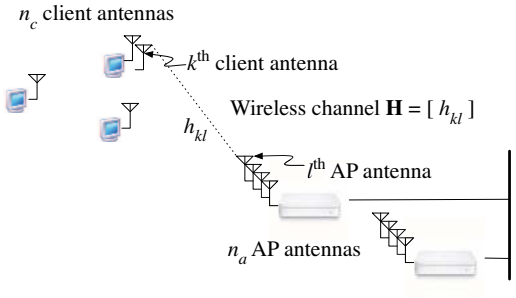
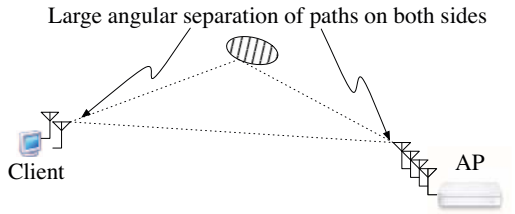


Figure 1: A MIMO wireless LAN with n_c client antennas and n_a AP antennas. Different clients and APs may transmit simultaneously, forming a MIMO system described by the channel matrix $\mathbf{H}$, whose entries characterize the wireless channel between a client's antenna and an AP's antenna.

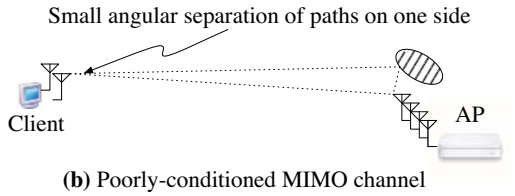
The emergence of new applications for wireless networks such as video internet telephony, data backup, and wearables like Google Glass, is today shifting the ratio between uplink and downlink traffic towards the former. In an uplink multi-user MIMO setting, clients may simply send their own information 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, each of the n_a receiving access point antennas then hears a mixture ($\mathbf{y}$) of signals sent from all n_c transmit antennas ($\mathbf{x}$), where $\mathbf{y} = \mathbf{Hx} + \mathbf{w}$ with $\mathbf{H}$ being a matrix whose entries describe the channel between each client and AP antenna and $\mathbf{w}$ representing background noise. Then, the *capacity* for a specific signal to noise ratio (SNR) is [59]

$$C = \mathrm{E}\left[\log\det\left(\mathbf{I}_{n_a} + \frac{\mathrm{SNR}}{n_c}\mathbf{H}\mathbf{H}^*\right)\right] \text{ bits/s/Hz.}$$

The question this paper considers is how to best turn this theoretical capacity into a practical throughput gain.



(a) Well-conditioned MIMO channel



(b) Poorly-conditioned MIMO channel

Figure 2: When reflectors are located solely in the vicinity of one of the endpoints 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 **H**.

The most frequent answer to this question is a demodulation scheme known as *zero-forcing*. In order to decouple interfering streams a zero-forcing receiver left-multiplies the received vector **y** with the inverse of matrix **H**, denoted $\mathbf{H}^{-1}$:

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

Zero-forcing has been proposed as part of a way to accomplish spatial division multiplexing in SAM [56] and BigStation [67], null out aligned interference in IAC [21], and enable concurrent 802.11n transmissions in 802.11n$^+$ [35]. Together with a strategy for choosing which users will transmit together (*user selection*), it works well in these systems, achieving multiplicative increases in throughput, in line with expectations. But can spatial multiplexing systems achieve even greater throughput?

The *condition number* of **H**, $\kappa(\mathbf{H})$, answers this question in the positive. The condition number is a well-known metric from numeric linear algebra that measures the sensitivity of the system **H** to noise [33]. When the wireless channel to the access point is favorable as shown in Figure 2(a), the matrix **H** is well-conditioned (small $\kappa(\mathbf{H})$), and the noise term $\mathbf{H}^{-1}\mathbf{w}$ above is small. But when reflectors are located solely in the vicinity of one of the endpoints as shown in Figure 2(b), $\kappa(\mathbf{H})$ becomes large [59], and its matrix determinant small. Under these conditions, when a zero-forcing receiver multiplies what it hears by the inverted channel, it amplifies background noise and interference **w**, leading to bit errors and decreased throughput [12, 49].

While the theory above is well-established, the following open experimental question immediately arises. How often is the indoor MIMO channel poorly-conditioned? An experimental characterization of our indoor wireless testbed (§5.1) and prior experimental wireless studies outdoors [32] both show that zero-forcing systems will experience an SNR reduction of 10 dB on approximately 20% of 2×2 MIMO channels and one half of all 4×4 MIMO channels, causing a significant throughput decrease.

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

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, and so is just now becoming practical for real systems. For example, in 2005, a 4×4 MIMO 16-QAM ASIC implementation achieved line rate over a 10 MHz frequency bandwidth [10], and then in 2012, a 4×4 MIMO 64-QAM ASIC implementation achieved 4G LTE line rates [64].

While the sphere decoder is practical today at the above rates, the search for higher throughputs is driving the use of even 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 [23, 42]. However, the challenge in adopting even denser constellations is that the sphere decoder's tree search has a branching factor equal to the size of the wireless constellation. This means that denser constellations necessitate a larger search space, with a corresponding increase in computation, overwhelming even the current state-of-the-art implementations mentioned above.[1] The challenge we tackle in this paper is to devise new tree traversal and tree pruning strategies that reduce the computational complexity the sphere decoder requires even with very dense constellations.

Paper contributions. Geosphere is a system for both multi-user and traditional[2] MIMO that incorporates two novel ways of reasoning about the geometry of signal constellations into the sphere decoder. Geosphere first: (a) contributes a new algorithm for determining the sphere decoder's search order (§3.1.1) and second (b) introduces a very efficient algorithm to prune the sphere decoder's search (§3.2) process, without sacrificing throughput. Geosphere is, to the best of our knowledge, the first system that makes 4×4 MIMO, 256-QAM communication practical. The paper also contributes the first, to our knowledge, rigorous experimental measurements of the conditioning of indoor MIMO wireless channels (§5.1).

Roadmap. 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 [43]. Here we show that Geosphere achieves substantial throughput gains over systems employing a combination of either zero-forcing or minimum mean-squared error successive interference cancellation (MMSE-SIC) decoding and user selection, and that in spatial-division multiplexing systems where many single-antenna clients transmit at the same time, Geosphere increases the per-user throughput. Results from our testbed show that Geosphere achieves average throughput gains of $2\times$ in 4×4 MIMO systems and 47% in 2×2 systems, while simultaneously requiring close to an order of magnitude less computation relative to the best known sphere decoder, bringing its computational demands in line with current systems already realized in ASIC and making SDR implementation possible. We then survey related work (§6) and conclude (§7).

2. PRIMER: THE SPHERE DECODER

This section provides essential background on the sphere decoder [2, 17], an algorithm able to determine the most-likely transmitted bits **x**

[1]For a 4×4 MIMO, 16-QAM system the sphere decoding tree has 6.6×10^4 nodes, while for 256-QAM it has 4.3×10^9 nodes.

[2]Geosphere also applies to the downlink, when both the AP and a single client are equipped with multiple antennas (*e.g.*, smart TVs).

in the MIMO system. 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

$$\mathbf{x}^* = \arg\min_{\mathbf{s}\in\mathcal{O}^{n_c}} \|\mathbf{y} - \mathbf{Hs}\|^2 . \quad (1)$$

This is the solution that minimizes detection errors and therefore maximizes throughput. Unfortunately, the computational complexity of the exhaustive search in Equation 1 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 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*:

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

Most sphere decoders begin with $r \leftarrow \infty$, and on finding a solution at distance $r' < r$ can safely set r to r' without the possibility of excluding the maximum-likelihood solution.

2.2 The tree

The sphere decoder recasts the maximum-likelihood problem (Equation 1) into a search in a tree of height n_c (number of client antennas) and branching factor $|\mathcal{O}|$ (constellation size). Figure 3 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 [53] 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 as

$$\widehat{\mathbf{y}} = \mathbf{Rs} + \mathbf{Q}^*\mathbf{w}, \text{where } \widehat{\mathbf{y}} = \mathbf{Q}^*\mathbf{y}, \quad (3)$$

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

$$d(\mathbf{s}) = K + \|\widehat{\mathbf{y}} - \mathbf{Rs}\|^2 . \quad (4)$$

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*

$$c(\mathbf{s}^{(l)}) = \left| \widehat{y}_l - \sum_{j=l}^{n_c} r_{lj} s_j \right|^2 . \quad (5)$$

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

$$d(\mathbf{s}^{(l)}) = d(\mathbf{s}^{(l+1)}) + c(\mathbf{s}^{(i)}). \quad (6)$$

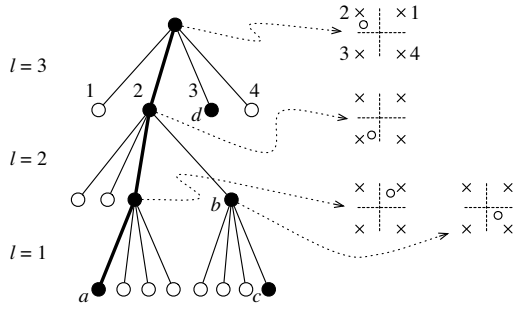


Figure 3: The sphere decoder operating on $n_c = 3$ transmit antennas, each sending a QPSK ($|\mathcal{O}| = 4$) symbol. Constellation points (denoted $\times$) and corresponding branches of the tree are numbered at the uppermost level ($l = 3$), and the received signal is denoted $\circ$. Visited nodes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

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

$$d(\mathbf{s}^{(l)}) \geq r^2, \quad (7)$$

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.

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 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 [46] after its inventors.

Continuing our example of Figure 3, 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 or siblings (note that the nodes are sorted) 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

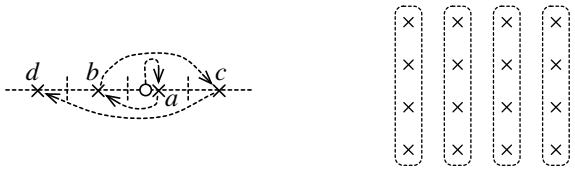


Figure 4: *Left:* The zigzag technique in a one-dimensional (PAM) constellation visits constellation points (×) in increasing distance from the received symbol (○). *Right:* Dividing a 16-QAM constellation into four 4-PAM subconstellations.

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 an improved, novel, pruning technique (§3.2). Later (§5), we experimentally evaluate the relative gains of each to highlight the different roles the two techniques play under varying channel conditions.

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 tree shown in Figure 3. 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

$$c\left(\mathbf{s}^{(l)}\right) = |r_{ll}|^2 \left|\widetilde{\mathbf{y}}_l - s_l\right|^2 \tag{8}$$

(where $\widetilde{\mathbf{y}}_l = \frac{\widehat{\mathbf{y}}_l - \sum_{j=l+1}^{n_c} r_{lj} s_j}{r_{ll}}$), 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 4 *(left)* shows a PAM constellation comprised of four constellation points (×) and a received symbol (○). 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)*.

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 ○. One inexact way of accomplishing this would be to partition the QAM constellation into PAM subconstellations as shown in Figure 4 *(left)*, and then zigzag "vertically" within each subconstellation. But this approach neglects the *in-phase* component of the received symbol.

Algorithm (two-dimensional zigzag)

1. Initialize a sorted priority queue $Q = \emptyset$, comprising constellation points (maintain Q sorted by Euclidean distance to ○ at all times).
2. Find the closest constellation point a to the received symbol by slicing ○ on the constellation's decision boundaries. constellation's decision boundaries). Calculate a's Euclidean distance and enqueue $a \rightarrow Q$.
3. Dequeue $Q \rightarrow x$ and explore x's children in the sphere decoder.
   (a) Zigzag vertically from x with respect to ○: call the result z_v. Calculate z_v's Euclidean distance to ○ and enqueue $z_v \rightarrow Q$.
   (b) Zigzag horizontally from x with respect to ○; 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 ○ and enqueue $z_h \rightarrow Q$.
4. Go to Step 3.

Figure 5: Two-dimensional zigzag algorithm pseudocode.

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 5 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 later in the experimental evaluation).

Example. Figure 6 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 (×) alongside the received symbol (○), above the priority queue Q. In Step *(i)*, the 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 ○, 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 the received symbol (the closest to ○) in Step *(v)*. Subsequent steps continue in the same manner, filling in the "expanding ring."

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.

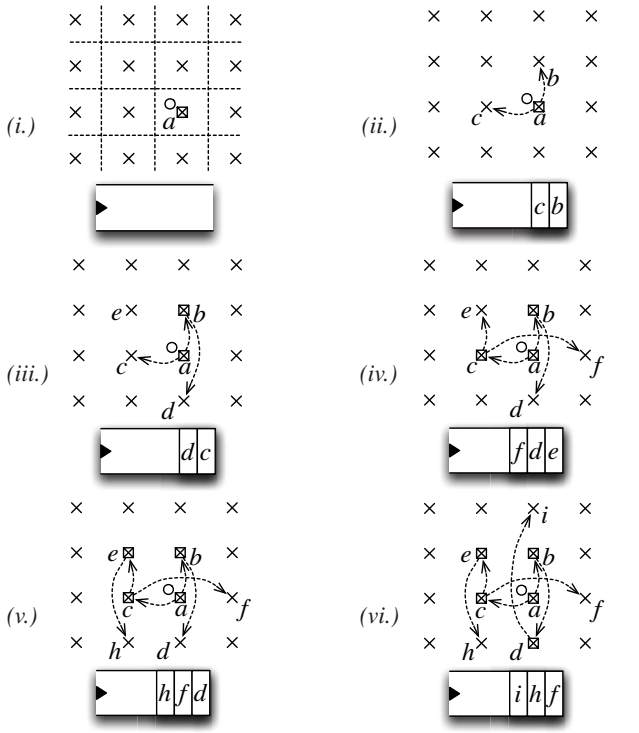


Figure 6: Geosphere's two-dimensional zigzag enumeration in the 16-QAM constellation. We denote constellation points $\times$, label points whose partial Euclidean distances have been computed, and denote points that have been explored $\boxtimes$.

Suppose that the sphere decoder has identified a currently-best candidate node a (referring to Figure 7) 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 multiplication operations and an addition.

Geosphere instead uses the constellation's geometry to establish a lower-bound on the exact branch cost, as shown in Figure 7. 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

$$\hat{c}\left(\mathbf{s}^l\right) = \sqrt{(2d_I - 1)^2 + (2d_Q - 1)^2} \tag{9}$$

based on a fast table lookup indexed on $|d_I|$ and $|d_Q|$ and uses $\hat{c}(\cdot)$ instead of $c(\cdot)$ in its pruning decision. Since $\hat{c}(\cdot) \leq c(\cdot)$, pruning based on Equation (9) alone doesn't exclude the maximum-likelihood solution, but may expand more branches than pruning based on exact branch costs. Thus, if geometrical pruning 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

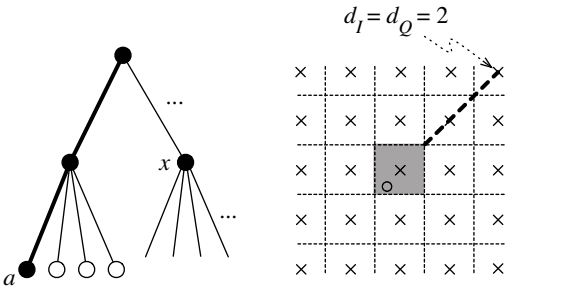


Figure 7: Geometrically lower-bounding the distance between received symbol $\circ$ and another constellation point at horizontal (d_I) and vertical (d_Q) offset two from the closest constellation point. Constellation points are spaced two units apart.

to 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 computational 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 zero-forcing systems that attempt to intelligently adapt to poorly-conditioned MIMO channels by varying the number of antennas and spatial streams they use. Finally, we evaluate Geosphere's computational complexity, comparing it with the well established ETH depth-first sphere decoder [10], one the few efficient sphere decoders able to achieve maximum-likelihood performance. Table 1 summarizes the experimental results presented here.

Testbed setup. Our testbed consists of single-antenna clients and four-antenna APs, communicating over a 20 MHz wireless channel in the 5 GHz ISM band. The distance between consecutive AP antennas is about 20 cm (approximately 3.2λ, where λ is the wireless wavelength) so that the wireless channels from each AP antenna to a client are uncorrelated with each other, and so representative of antenna spacings above λ indoors at 5 GHz [31].

We evaluate Geosphere in actual office conditions: Figure 8 shows the testbed environment, including the places we position APs and clients. Note that the topology includes both line-of-sight and non-line-of-sight 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 experimental 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 MIMO channels corresponding to 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 8 shows their positions, with hollow circles and red triangles denoting client positions in this experiment, and squares denoting APs.

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× throughput gains over multi-user MIMO for four AP antennas and four clients, (47% gain in 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 ([25], *cf.* §5.3), making the sphere decoder practical for dense constellations.

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

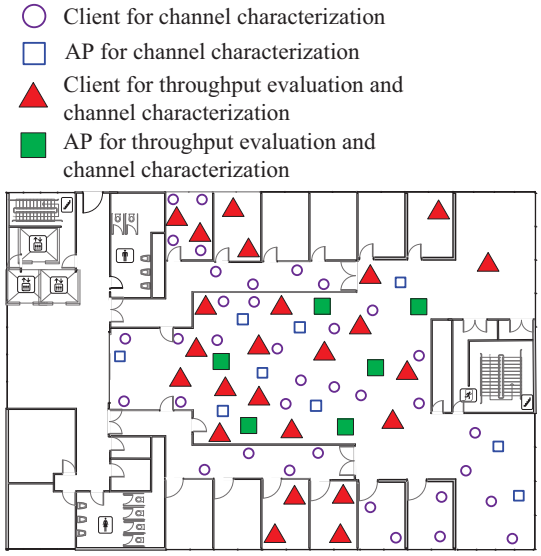


Figure 8: Floor plan of the office space housing the wireless testbed used in Geosphere's experimental evaluation.

In order to characterize the channel we will use two metrics. The first is the square of the condition number $\kappa^2(\mathbf{H})$ which is a good upper-bound on the actual noise amplification due to zero-forcing [33]. However, this metric doesn't necessarily provide us with the actual performance difference between a zero-forcing system and one that uses 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 shown 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 9 and Figure 10 we show 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 10). We observe that the use of zero-forcing will

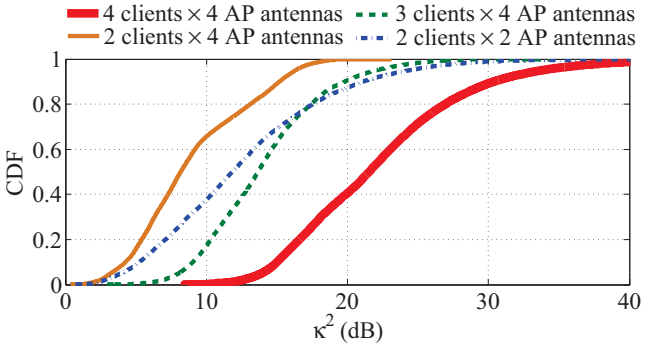


Figure 9: Cumulative distribution across testbed links, OFDM subcarriers, and spatial streams of κ^2 (decibels), the power of the MIMO channel condition number. Higher values of κ^2 indicate worse channel conditioning.

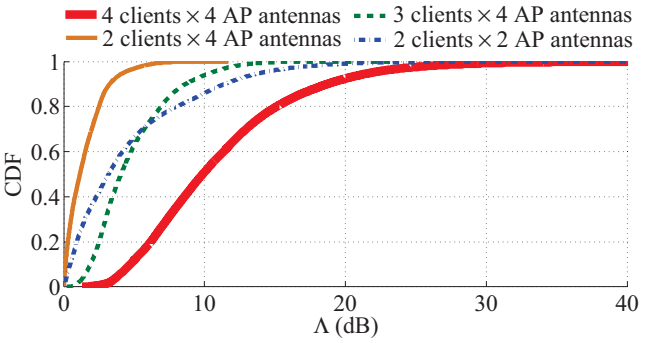


Figure 10: Cumulative distribution across testbed links and OFDM subcarriers of Λ, the SNR ratio degradation that the most-degraded user experiences for each particular link.

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 [67] do. From Figures 9 and 10, 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, referring to the "2 clients × 4 AP antennas" curve in Figure 10, if only two clients transmit, the maximum degradation due to zero-forcing will be less than three decibels for 90% of the channels. That means we could sacrifice concurrency to reduce the degradation due to zero-forcing.

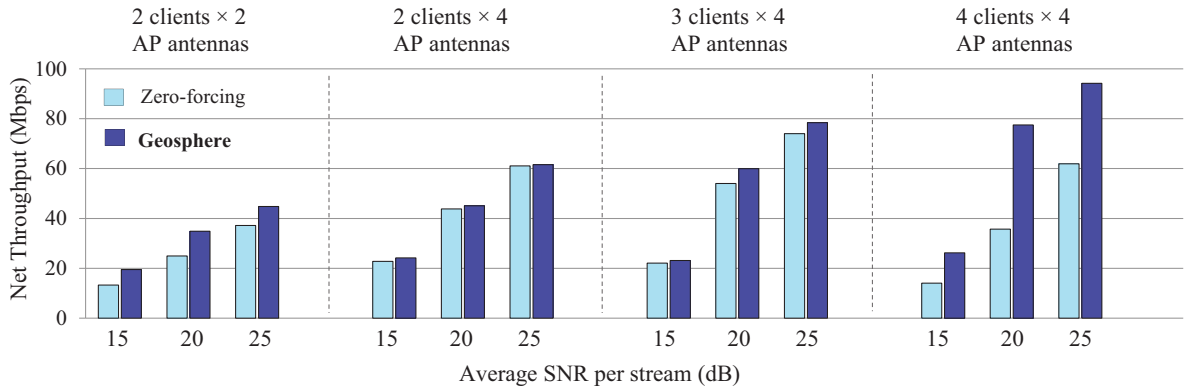


Figure 11: Experimental testbed throughput comparison between zero-forcing MIMO and Geosphere for different numbers of clients, number of AP antennas and SNRs.

Would this be a net benefit to throughput? We next examine this question in close detail.

5.2 System throughput

We now measure the uplink throughput that zero-forcing 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; we now test 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 8. We send data to the APs using various modulations to characterize performance at different transmission rates: we transmit 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 $\pm$5 dB, 20 dB $\pm$ 5 dB, and 25 dB $\pm$5 dB, where the quoted SNR is the average SNR over all transmitted streams. Selecting users in a small SNR range around a specific value is a practical user selection method to keep the condition number small. Larger gains are expected for Geosphere's if the users are selected randomly. In addition, in lieu of implementing a rate adaptation algorithm, we show throughput results for the constellation that achieves the best average throughput for the corresponding range; this emulates ideal bit rate adaptation and makes the results independent of the rate adaptation method employed.

Results. In Figure 11 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, Geosphere's 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 most challenging case of two and three clients and an AP with four receive antennas (where channels are most often well-conditioned) Geosphere provides average gains of 6%. These throughput gains are consistent with what we expect 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 11 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 SNR).

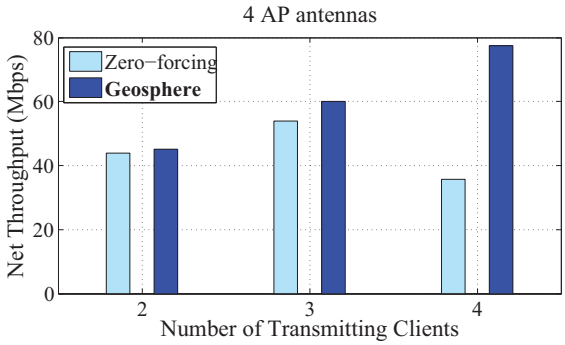


Figure 12: Experimental testbed throughput comparison between zero-forcing MIMO and Geosphere for different numbers of users accessing a four-antenna AP at the same time, at 20 dB SNR.

Figure 12 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. Therefore, with Geosphere we can increase the number of clients while keeping the throughput of each client unaffected, which is not feasible with zero-forcing.

5.2.1 Comparison with MMSE-SIC detection

The same effect can be seen in Figure 13, where we simulate a ten-antenna AP with different numbers of clients, again at 20 dB SNR. In these experiments we also consider MMSE-SIC receiver processing which orders users by descending SNR, then performs MMSE detection and interference cancellation successively for each user, an approach known to be capable of reaching multi-user capacity [59].

In Figure 13 we see that as long as we operate far from the maximum achievable throughput and only a limited number of clients are transmitting, all methods have similar performance. However,

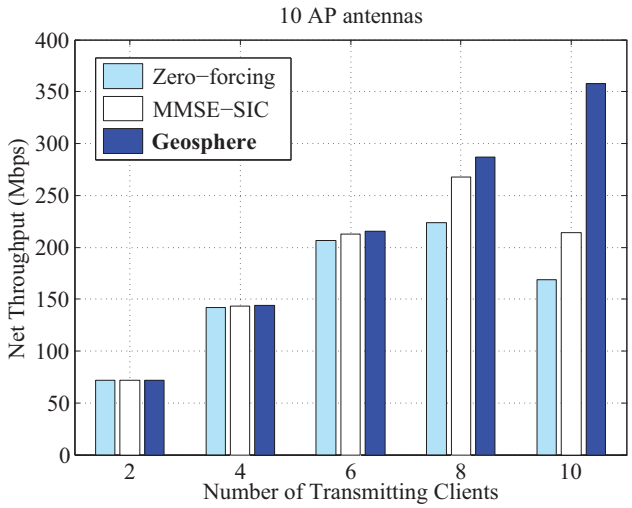


Figure 13: Simulation-based throughput comparison between zero-forcing MIMO, MMSE-SIC and Geosphere for different numbers of users accessing a ten-antenna AP at the same time over a Rayleigh fading channel.

for numbers of clients similar to the number of antennas, where the throughput becomes maximum, the performance difference between Geosphere and the other approaches increases, and Geosphere is almost two times faster for the 10×10 case.[3] We can also see that MMSE-SIC significantly outperforms zero-forcing, but despite its good information theoretical properties, in practice, it cannot optimize throughput due to error-propagation. In addition, SIC methods have a significant drawback compared to zero-forcing and Geosphere that limit their practicality. The decoding of the different users needs to be performed sequentially which linearly increases decoding latency.

5.3 Computational complexity

We now quantify the computation Geosphere requires. To this end, we compare Geosphere against the most efficient known depth-first sphere decoder implementation able to achieve maximum-likelihood performance; we denote this system *ETH-SD* in the following experimental results.

We base our implementation of ETH-SD on the VLSI implementation of Burg *et al.* [10], but instead of decomposing the constellation into a set of constant-amplitude phase-shift keying subconstellations as they do, we use the superior method of Hess *et al.* [25]. To determine the node to visit next, Hess' method splits the QAM constellation into horizontal subconstellations, performs an one-dimensional zigzag, and then compares Euclidean distances across all subconstellations. This approach is more efficient since for dense constellations it involves partitioning into fewer sub-constellations, hence is a more challenging comparison design point for Geosphere.

One frequently-used measure of computational complexity in the literature is the number of visited nodes in the sphere decoder tree. However since Geosphere performs some additional computation to avoid visiting nodes, we require a metric that captures this additional computation. Since the dominant part of the additional computation is partial Euclidean distance calculations, this metric tracks overall complexity accurately, and so we primarily use this metric in our evaluation, as is also common in the literature [40]. For completeness and additional insight into why Geosphere improves performance, we also report number of visited nodes.

Since in an OFDM system, MIMO processing takes place on a per subcarrier basis, we report the preceding metrics as needed per subcarrier, averaged across all subcarriers.

5.3.1 Testbed-based complexity evaluation

Methodology. First we compare the complexity of Geosphere and ETH-SD for the live testbed experiments of the previous subsection. These complexity results measure the corresponding amount of computation required to obtain the throughput results we show in Figure 11.

Results. In Figure 14 we show the average number of partial Euclidean distance calculations for all experiments. We see that Geosphere is consistently less computationally demanding 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 computational savings can be up to 63%.

As noted above, the throughput gains of Geosphere are modest for well-conditioned channels. One might therefore be tempted to argue in favor of a system that switches back to zero-forcing when faced with a well-conditioned wireless channel. However, the above results show that Geosphere actually adjusts its computational complexity to the current SNR, and so complexity at high SNR is actually very small, obviating the need for a hybrid system.

5.3.2 Simulation-based complexity evaluation

We now quantify the computation Geosphere requires with the purpose of convincing the reader that our system can achieve 256-QAM, 4×4 MIMO performance with a computational demand on par with 64-QAM sphere decoders currently implemented in ASIC. We also break down the complexity of each of Geosphere's two main components: two-dimensional zigzag enumeration (§3.1.1), and geometrical pruning (§3.2).

Methodology. Since the WARP platform's analog front end limits it to a maximum SNR of approximately 30 dB over the links in our testbed, for the following computational complexity experiments we perform simulations. To analyze the source of Geosphere's gains, we run the following two variants of our system:

1. *2D zigzag only*: Geosphere, running two-dimensional zigzag enumeration sorting without geometrical pruning.
2. *Full*: Geosphere's full design, including two-dimensional zigzag enumeration sorting and geometrical pruning.

We present both ***(a)*** trace-based simulation, driven by empirical MIMO channel measurements collected from our WARP testbed, and ***(b)*** simulation over a MIMO Rayleigh fading channel with independent, identically-distributed channel realizations sampled on a per-frame basis.[4]

Results. In Figure 15 we show complexity for an SNR such that each constellation reaches a frame error rate of approximately 10% (*e.g.*, approximately 27, 33 and 39 dB for the 2×4 measured channels and 16-, 64- and 256-QAM constellations, respectively). We examine two MIMO cases: In Figure 15(a) we show complexity for two clients and four AP antennas. In this case, complexity is relatively low, due to favorable MIMO channel conditioning, but at the cost of reduced throughput, since only two users transmit. We note that the complexity of ETH-SD increases with constellation

[3]The authors of BigStation [67] note a similar trend in their experimental results.

[4]This accurately quantifies performance over channels whose coherence times are greater than the time for one frame, *i.e.* driving speeds and slower.

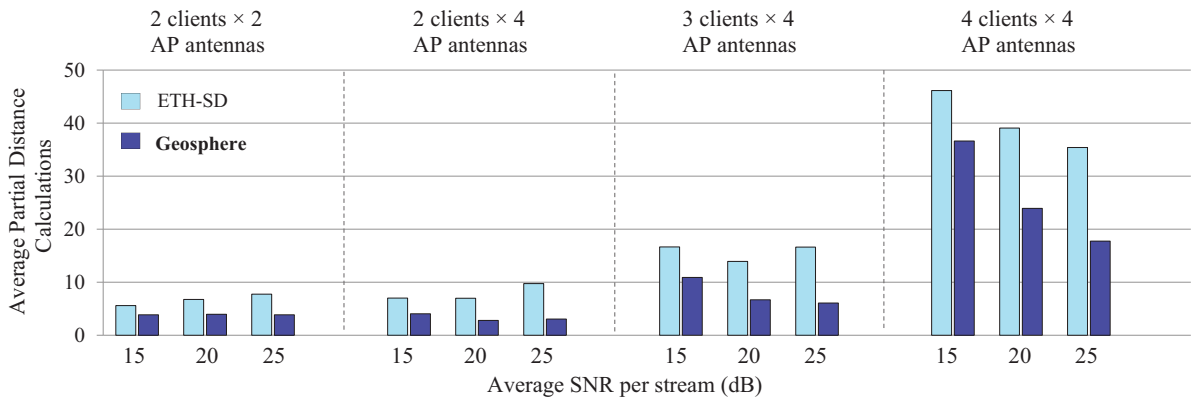


Figure 14: Complexity comparison between ETH-SD and Geosphere for different numbers of clients and AP antennas.

size, while the complexity of Geosphere is substantially smaller, independent of the constellation size, and comparable to the complexity of zero-forcing.[5] For the Rayleigh channel, Geosphere is 81% less complex than ETH-SD for the 256-QAM case, while the full Geosphere (with 2D zigzag and geometrical pruning) provides complexity gains of 27% compared to 2D-zigzag-only Geosphere.

Figure 15(b) shows complexity for four clients and four AP antennas, where we need to cope with more challenging MIMO channel conditions. For the 4×4 case we see that the complexity of ETH-SD (but not Geosphere) greatly increases with constellation size. As a result Geosphere is up to 70% less complex than ETH-SD for the Rayleigh channel. In addition we see that the zigzag algorithm is the main source of complexity improvement for large constellations, while early pruning provides complexity gains of 13–17%.

Discussion. In general, the effect of geometrical pruning becomes more apparent for better SNRs and channel conditions. For example, in any depth-first sphere decoder, we need n_t partial Euclidean distance calculations to find the distance of the first candidate solution (first leaf). If this is the correct solution and it has a small Euclidean distance (as in the case of high SNR), a typical sphere decoder would require at least another $n_t - 1$ partial distance calculations to prune the rest of the tree. In contrast, geometrical pruning prunes the rest of the tree without any additional calculation. Therefore, if in the simulations above, we increase the SNR to reach target packet error rates of 1%, geometrical pruning reaches a 47% improvement compared to Geosphere with zigzag only.

Another significant characteristic of Geosphere is that since it and all leading practical sphere decoders (including ETH-SD) use the Schnorr-Euchner enumeration (§6), the number of visited nodes is the same for all of them. Therefore, Geosphere maintains the processing throughput of hardware architectures that process one node per clock cycle [10].

Finally, we also observe that while the collected channels are not Rayleigh distributed, the complexity results are in very good agreement. Empirically, this suggests that only the target bit error rate (and not channel fading statistics nor the operating SNR) determines the amount of computation required to decode.

6. RELATED WORK

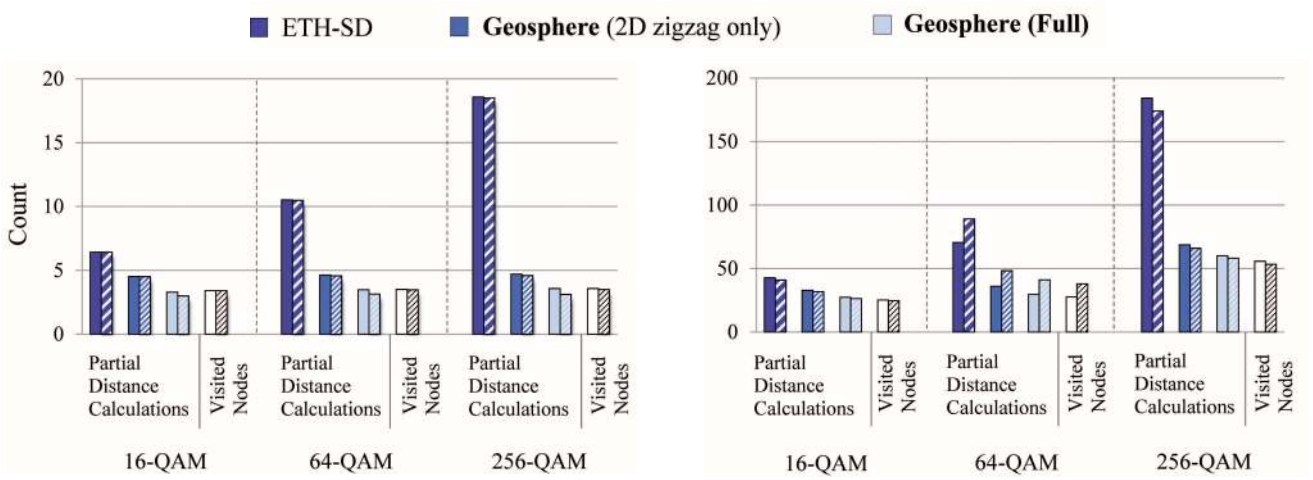
Work on the sphere decoder has been extensive, and we are not the first to note the importance of and experimentally measure the condition number of the MIMO channel, or propose solutions for noise amplification. Here we discuss related work in these areas, in both the networked systems and communications theory communities, placing Geosphere into context and highlighting our contributions.

Linear filtering. We note that the classical 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 [59]. Artés *et al.* [3] 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, but their method does not maintain performance in larger constellations. Leveraging the power of the sphere decoder, Geosphere maintains performance while scaling to 256-QAM.

SIC. Successive interference cancellation (SIC) methods decode signals from the strongest interferers and subtract their effect on the other signals [69]. However, the efficacy of SIC is contingent on the information transmitted between users being sent at a rate that allows correct detection in the presence of uncancelled interference, a requirement which does not hold for a sphere decoders. Sayana *et al.* [45] use successive interference cancellation [59] 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, because it operates under the coding layer.

Scheduling and user selection. When the number of users is very large, zero-forcing combined with a user-selection strategy becomes asymptotically optimal [61]. Yoo and Goldsmith note the optimality of zero-forcing beamforming and propose a scheduling algorithm for a large number of users [68]. Chen and Wang analyze the interaction of zero-forcing and time-division scheduling techniques, proposing to select and schedule users so that noise amplification due to zero-forcing is lessened [12]. However, such approaches require a number of clients needing to send data that is *orders of magnitude* greater than the number of AP antennas, making them of limited applicability in a common case where a small number of clients (less than 10) saturate the wireless medium. With extremely large user numbers, the process of estimating and tracking the wireless channel to each also incurs non-negligible overhead. Furthermore,

[5] Zero-forcing requires $n_t \times n_r = 8$ complex multiplications, whereas Geosphere requires at most 10 complex multiplications (assuming that each partial distance calculation requires $n_t + 1$ multiplications).



(a) Two clients and four AP antennas.

(b) Four clients and four AP antennas.

Figure 15: Simulation-based complexity comparison between ETH-SD and Geosphere. ***Solid bars:*** Simulated Rayleigh channel; ***striped bars:*** empirically measured channel. N.B.: each of the above sphere decoders visit the same number of nodes.

we have experimentally shown that Geosphere improves throughput consistently for both small and middling numbers of clients.

6.1 Sphere decoder optimizations

There is a large body of prior work that optimizes the performance of the sphere decoder. It roughly breaks down into proposals that simplify the Euclidean distance calculation (discussed next), optimize the order of visiting nodes, and prune the tree more aggressively.

Simplified distance measures. These proposals approximate Euclidean distance in the constellation with computationally simpler but less precise distance measures, to incrementally lower processing overhead [10, 25, 62]. However, such approaches increase the bit error probability while remaining impractical when sending dense constellations, and so Geosphere outperforms these approaches.

Node ordering-based optimizations. Chan and Lee [11] first proposed the frequently used radius update approach. However, their proposal doubles the height of the tree, making it impractical for implementation [10]. Zhao and Giannakis [70] generalize Schnorr-Euchner enumeration probabilistically, 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 common SNR range. Ghasemmehdi and Agrell [20] minimize the number of visited nodes, but also double tree height and require impractical amounts of memory.

Some prior work takes constellation geometry into account. Mennenga and Fettweis [37] propose an enumeration method that geometrically splits the I-Q space into sectors. However, their algorithm can accurately sort only the first eight nodes, resulting in decode errors and decreased throughput. As part of a K-best sphere decoder, Shabany *et al.* [48] propose an enumeration method superficially similar to Geosphere's two-dimensional zigzag. However, Geosphere's algorithm is superior, enumerating in Step 3(b) only if no other constellation point in z_h's PAM subconstellation is in Q, yielding a significant reduction in partial distance calculations for dense constellations and depth-first sphere decoding. For example, when expanding a node to identify the child with the third smallest Euclidean distance, Geosphere needs four partial distance calculations while Shabany's needs five (25% more).

Pruning-based optimizations. Many proposals attempt to reduce complexity by probabilistically pruning tree branches unlikely to survive pruning. Work by Shim and Kang [50, 51] doubles decoder tree height and requires cumbersome tuning to tradeoff complexity with performance. Cui *et al.* propose statistical node pruning strategies [14, 15], but incur a significant loss of performance in order to achieve non-negligible complexity gains, making their proposals unsuitable for practical use. Stojnic *et al.* propose a pruning technique that requires solving a semi-definite problem on top of the usual sphere decoder search tree [52], but their technique is only appropriate for very low SNR (less than 6 dB), whereas Geosphere targets relatively higher SNR ranges and larger numbers of users. Gowaikar and Hassibi propose a related probabilistic pruning technique [22], but it achieves worst performance than the preceding proposal of Shim *et al.* [50].

Breadth-first sphere decoders. In contrast to depth-first sphere decoders, breadth-first sphere decoders have average complexity typically higher than depth-first approaches [10]. The fixed-complexity sphere decoder [5] 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 maximum-likelihood performance at high SNRs [30], with higher computational complexity than a depth-first approaches. Geosphere, on the other hand, reaches maximum-likelihood performance while significantly reducing computational complexity.

K-best sphere decoders. K-best sphere decoders [13, 24, 34, 39, 47, 48, 63, 66] are depth-first sphere decoders that select the K best branches at each level of the tree regardless of the sphere constraint or any other distance control policy. However, the choice of K is speculative and increases with the order of the constellation, making K-best inappropriate for dense constellations. Furthermore, K must be increased to accommodate the worst MIMO channel, making such schemes inefficient. To reduce complexity and increase parallelizability, Azzam and Ayanoglou [4] propose to reorder the

channel matrix, but double its size (and thus the height of the sphere decoder tree) to accommodate practical QAM channels. Thus, the required complexity is is still very high.

Channel condition-aware sphere decoders. Maurer *et al.* propose a system that switches between zero-forcing and maximum-likelihood decoding via a threshold test on the channel condition number [36]. 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.* [44] propose a sphere decoder that expands at most K branches of each node in the decoding tree, varying K based on $\kappa(\mathbf{H})$. Geosphere alleviates the need for such complex designs since it can automatically adjust its complexity to the condition of the channel, and can reduce it down complexities similar to zero-forcing. In addition, we present a full working system design and experimental evaluation in real indoor office conditions.

Su and Wassel [55] use a geometrically-inspired ordering of the MIMO channel matrix $\mathbf{H}$ before performing sphere decoding. However, the resulting computational savings vanish for average and high SNR values of practical interest.

The generalized sphere decoder. Generalized sphere decoders [16, 18, 19] are designed for situations where the number of transmit antennas exceeds the number of receive antennas, and so the MIMO channel matrix is rank deficient (as opposed to merely poorly-conditioned). However, these techniques don't increase capacity, since capacity increases with $\min\{n_a, n_c\}$. But since these techniques use Schnorr-Euchner enumeration, Geosphere improves throughput and computational complexity synergistically.

6.2 MIMO channel condition measurements

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.* [57], in the 2.16–2.18 GHz frequency band), indoors, but in a mobile cellular frequency band (*e.g.*, Kita *et al.* [32]), 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 comparable to our measurements (§1, §5), suggesting that the problem of poor channel conditioning occurs in general, in outdoor as well as indoor environments, and across the range of microwave frequencies.

Channel hardening [26, 29] 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 achieve linear throughput gains.

6.3 Downlink beamforming

In the downlink, sphere decoder-based techniques can be used at the transmitter in lieu of zero-forcing based precoding; this is known as *sphere encoder precoding* [6, 27, 38, 41]. This precoding, however, requires that APs track the wireless channel as they move, which adds complexity and becomes harder with increasing mobility. Nonetheless, since Geosphere's techniques are receiver-based, Geosphere is complementary to precoding: the two can achieve complementary performance gains if implemented together.

6.4 System designs

The Spinal codes [42] 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 they may be extended to the MIMO channel in the future.

The authors of BigStation [67] 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 using many antennas and radios (with their associated costs) at the AP.

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 real high-rate wireless systems (using dense constellations) with a geometrical approach based on soft information.

While Geosphere increases throughput, iterative soft receiver processing is required to reach MIMO capacity [28]. Such "soft-detectors" consist of several constrained maximum-likelihood problems and therefore the sphere decoder can be of use [9, 60]. State-of-the-art soft-input, soft-input sphere decoders [7, 8, 54, 65] are based on the ETH-SD approach, but their complexity remains prohibitive under dense constellations. Since Geosphere outperforms ETH-SD, a promising next step is to extend our techniques to this setting.

Acknowledgements: The research leading to these results has received funding from the European Research Council under the EU's Seventh Framework Programme (FP/2007-2013), ERC Grant Agreement n° 279976. We thank our shepherd Shyamnath Gollakota and the anonymous reviewers for their insightful feedback.

8. REFERENCES

[1] Agilent Technologies. MIMO Performance and Condition Number in LTE Test: App. 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. *IEEE Tr. Sig. Proc.*, 51(11):2808–20, 2003.
[4] L. Azzam and E. Ayanoglu. Reduced complexity sphere decoding via a reordered lattice representation. *IEEE Tr. Comms.*, 57(9):2564–69, 2009.
[5] L. Barbero and J. Thompson. Fixing the complexity of the sphere decoder for MIMO detection. *IEEE Trans. on Wireless Comms.*, 7(6):2131–2142, 2008.
[6] M. Barrenechea et al. Implementation of complex enumeration for multiuser MIMO vector precoding. In *Proc. of Eur. Sig. Proc. Conf.*, 2011.
[7] F. Borlenghi et al. A 772 Mbit/s 8.81 bit/nJ 90 nm CMOS soft-input soft-output sphere decoder. In *Proc. of IEEE Asian Solid State Circ. Conf.*, 2011.
[8] F. Borlenghi et al. A 2.78 mm^2 65 nm CMOS gigabit MIMO iterative detection and decoding receiver. In *Proc. of the ESSCIRC*, pages 65–68, 2012.
[9] J. Boutros et al. Soft-input soft-output lattice sphere decoder for linear channels. In *GLOBECOM*, 2003.
[10] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bolcskei. VLSI implementation of MIMO detection using the sphere decoding algorithm. *IEEE J. of Solid-State Circ.*, 40(7):1566–1577, 2005.
[11] A. Chan and I. Lee. A new reduced-complexity sphere decoder for multiple antenna sys. In *IEEE ICC*, 2002.

[12] C. Chen and L. 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.

[13] S. Chen, T. Zhang, and Y. Xin. Relaxed K-best MIMO signal detector design and VLSI implementation. *IEEE Trans. on VLSI Sys.*, 15(3):328–337, 2007.

[14] T. Cui, S. Han, and C. Tellambura. Probability distribution based node pruning for sphere decoding. *IEEE Trans. on Veh. Tech.*, 62(4):1586–96, 2013.

[15] T. Cui, T. Ho, and C. Tellambura. Statistical pruning for near maxiumum likelihood detection of MIMO systems. In *IEEE ICC*, 2007.

[16] T. Cui and C. Tellambura. An efficient generalized sphere decoder for rank-deficient MIMO systems. *IEEE Comms. L.*, 9(5):423–5, 2005.

[17] M. Damen, H. El Gamal, and G. Caire. On maximum likelihood detection and the search for the closest lattice point. *IEEE T. Inf. Th.*, 49(10):2389–402, 2003.

[18] M. Damen et al. Generalized sphere decoder for asymmetrical space-time communication architecture. *IEEE Elec. L.*, 36(2):166–67, 2000.

[19] P. Dayal and M. Varanasi. A fast generalized sphere decoder for optimum decoding of under-determined MIMO systems. In *Proc. of Allerton Conf.*, 2003.

[20] A. Ghasemmehdi and E. Agrell. Faster recursions in sphere decoding. *IEEE T. Inf. Th.*, 57(6):3530–6, 2011.

[21] S. Gollakota, S. Perli, and D. Katabi. Interference alignment and cancellation. In *SIGCOMM*, 2009.

[22] R. Gowaikar and B. Hassibi. Statistical pruning for near-maximum likelihood decoding. *IEEE Tr. on Sig. Proc.*, 55(6):2661–75, 2007.

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

[24] Z. Guo and P. Nilsson. Algorithm and implementation of the K-best sphere decoding for MIMO detection. *IEEE J. Sel. Areas Commun.*, 24(3):491–503, 2006.

[25] C. Hess et al. Reduced-complexity MIMO detector with close-to-ML error rate performance. In *ACM Great Lakes VLSI Symp.*, 2008.

[26] 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.

[27] B. Hochwald, C. Peel, and A. Swindlehurst. A vector-perturbation technique for near-capacity multiantenna multiuser communication. *IEEE Trans. on Comms.*, 53(3):537–544, 2005.

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

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

[30] J. Jaldén, L. Barbero, B. Ottersten, and J. Thompson. The error probability of the fixed-complexity sphere decoder. *IEEE Tr. on Sig. Proc.*, 57(7):2711–20, 2009.

[31] P. Kafle et al. Spatial correlation and capacity measurements for wideband MIMO channels in indoor office environment. *IEEE Trans. on Wireless Comms.*, 7(5):1560–1571, 2008.

[32] N. Kita et al. Measurement of Demel condition number for 2x2 MIMO-OFDM channels. In *IEEE VTC*, 2004.

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

[34] Q. Li and Z. Wang. Improved K-best sphere decoding algorithms for MIMO systems. In *Proc. of IEEE Int. Symp. on Circuits and Sys.*, 2006.

[35] K. Lin, S. Gollakota, and D. Katabi. Random access heterogeneous MIMO networks. In *SIGCOMM*, 2011.

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

[37] B. Mennenga and G. Fettweis. Search sequence determination for tree search based detection algorithms. In *IEEE Sarnoff Symp.*, 2009.

[38] M. Mohaisen and K. Chang. Fixed-complexity sphere encoder for multi-user MIMO systems. *Journal of Communications and Networks*, 13(1):63–69, 2011.

[39] S. Mondal et al. Design and implementation of a sort-free K-best sphere decoder. *IEEE Trans. on VLSI Sys.*, 18(10):1497–1501, 2010.

[40] K. Nikitopoulos et al. Complexity-efficient enumeration techniques for soft-input, soft-output sphere decoding. *IEEE Comms. L.*, 14(4):312–4, 2010.

[41] C. Peel et al. A vector-perturbation technique for near-capacity multiantenna multiuser communication. *IEEE Trans. on Comms.*, 53(1):195–202, 2005.

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

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

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

[45] 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.

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

[47] M. Shabany and P. Gulak. Scalable VLSI architecture for K-best lattice decoders. In *IEEE ISCAS*, 2008.

[48] M. Shabany, K. Su, and P. Gulak. A pipelined scalable high-throughput implementation of a near-ML K-best complex lattice decoder. In *IEEE ICASSP*, 2008.

[49] W. Shen et al. Rate adaptation for 802.11 multiuser MIMO networks. In *MobiCom*, 2012.

[50] B. Shim and I. Kang. Sphere decoding with a probabilistic tree pruning. *IEEE Trans. on Signal Processing*, 56(10):4867–4878, 2008.

[51] B. Shim and I. Kang. On further reduction of complexity in tree pruning based sphere search. *IEEE Trans. on Comms.*, 58(2):417–22, 2010.

[52] M. Stojnic et al. Further results on speeding up the sphere decoder. In *IEEE ICASSP*, 2006.

[53] G. Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 4th edition, 2009.

[54] C. Studer and H. Bölcskei. Soft-input soft-output sphere decoding. In *IEEE ISIT*, 2008.

[55] K. Su and I. Wassell. A new ordering for efficient sphere decoding. In *IEEE ICC*, 2005.

[56] K. Tan et al. SAM: Enabling practical spatial multiple access in wireless LAN. In *MobiCom*, 2009.

[57] H. Teague et al. Field results on MIMO performance in UMB systems. In *IEEE VTC*, 2008.

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

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

[60] H. Vikalo, B. Hassibi, and T. Kailath. Iterative decoding for MIMO channels via modified sphere decoding. *IEEE Trans. on Wireless Comms.*, 3(6):2299–2311, 2004.

[61] J. Wang et al. User selection with zero-forcing beamforming achieves the asymptotically optimal sum rate. *IEEE Tr. on Sig. Proc.*, 56(8):3713–26, 2008.

[62] M. Wenk, L. Bruderer, A. Burg, and C. Studer. Area-and throughput-optimized VLSI architecture of sphere decoding. In *EEE/IFIP VLSI-SOC*, 2010.

[63] M. Wenk et al. K-best MIMO detection VLSI architectures achieving up to 424 Mbps. In *ISCC*, 2006.

[64] M. Winter et al. A 335 Mb/s 3.9 mm^2 65 nm CMOS flexible MIMO detection-decoding engine achieving 4G wireless data rates. In *IEEE ISSCC*, 2012.

[65] E. Witte et al. A scalable VLSI architecture for SISO single tree-search sphere decoding. *IEEE Trans. on Circ. and Sys.*, 57(9):706–710, 2010.

[66] K. Wong et al. A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels. In *ISCC*, 2002.

[67] Q. Yang et al. BigStation: Enabling scalable real-time signal processing in large MU-MIMO systems. In *SIGCOMM*, 2013.

[68] T. Yoo and A. Goldsmith. On the optimality of multiantenna broadcast scheduling using zero-forcing beamforming. *IEEE JSAC*, 24(3):528–541, 2006.

[69] A. Zanella, M. Chiani, and M. Win. MMSE reception and successive interference cancellation for MIMO systems with high spectral efficiency. *IEEE Trans. on Wireless Comms.*, 4(3):1244–1253, 2005.

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